Microprocessors and Microsystems 51 (2017) 76–98

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

Optimization-based power and thermal management for dark silicon aware 3D chip multiprocessors using heterogeneous cache hierarchy

Arghavan Asad[a,*], Ozcan Ozturk[b], Mahmood Fathy[a], Mohammad Reza Jahed-Motlagh[a]

[a] Computer Engineering Department, Iran University of Science and Technology, Tehran, Iran
[b] Computer Engineering Department, Bilkent University, Ankara, Turkey

ARTICLE INFO

Article history:
Received 5 January 2016
Revised 29 December 2016
Accepted 27 March 2017
Available online 14 April 2017

Keywords:
Hybrid cache hierarchy
Reconfigurable cache
Non-volatile memory (NVM)
Three-dimensional integrated circuits
Dark-silicon
Chip-multiprocessor (CMP)
Network-on-chip (NoC)
Optimization

ABSTRACT

Management of a problem recently known as “dark silicon” is a new challenge in multicore designs. Prior innovative studies have addressed the dark silicon problem in the fields of power-efficient core design. However, addressing dark silicon challenges in uncore component designs such as cache hierarchy, on-chip interconnect etc. that consume significant portion of the on-chip power consumption is largely unexplored. In this paper, for the first time, we propose an integrated approach which considers the impact of power consumption of core and uncore components simultaneously to improve multi/many-core performance in the dark silicon era. The proposed approach dynamically (1) predicts the changing program behavior on each core; (2) re-determines frequency/voltage, cache capacity and technology in each level of the cache hierarchy based on the program's scalability in order to satisfy the power and temperature constraints. In the proposed architecture, for future chip-multiprocessors (CMPs), we exploit emerging technologies such as non-volatile memories (NVMs) and 3D techniques to combat dark silicon. Also, for the first time, we propose a detailed power model which is useful for future dark silicon CMPs power modeling. Experimental results on SPEC 2000/2006 benchmarks show that the proposed method improves throughput by about 54.3% and energy-delay product by about 61% on average, respectively, in comparison with the conventional CMP architecture with homogenous cache system.

(A preliminary short version of this work was presented in the 18th Euromicro Conference on Digital System Design (DSD), 2015.)

1. Introduction

Even though the development of semiconductor technology continues to provide increasing on-chip transistor densities and enabling the integration of many cores on a single die, Dennard scaling [1], which offer near-constant chip power with the doubling of transistors, has come to an end.

Due to the breakdown of Dennard scaling, the fraction of transistors that can be simultaneously powered on within the peak power and temperature budgets is dropping exponentially with each process generation. This phenomenon has been termed as the dark silicon era [2]. Predictions in literature indicate that if dark silicon challenges is not addressed properly, more than 90% of fractions of chips will be effectively dark, idle, dim, or under-clocked dark silicon, within 6 years [3]. Therefore, it is extremely important to provide next generation architectural techniques, design tools, and analytical models for future many-core CMPs in the presence of dark silicon [4].

In the nanometer era, leakage power depletes the power budget and has substantial contribution in overall power consumption. In this regard, study has shown that over 50% of the overall power dissipation in 65 nm generation is due to the leakage power [5] and this percentage is expected to increase in the next process generations [6,7]. Also, research shows that the increasing leakage power consumption is a major driver of unusable portion or dark silicon in future many-core CMPs [2].

In recent years, more and more applications are shifting from compute bounding to data bounding, thereby, a hierarchy of cache levels to efficiently store and manipulate large amounts of data is required. In this context, an increasing percentage of on-chip transistors are invested on the cache hierarchy and architects have dramatically increased the size of cache levels in cache hierarchy, in an attempt to bridge the gap between fast cores and slow off-chip memory accesses in multi/many-core CMPs. Considering the fact that cache hierarchy occupies as much as 50% of the chip area, it is dominant leakage consumer in future multi/many-core systems. Also, since leakage power has become a significant factor in

* Corresponding author.
E-mail address: ar_asad@comp.iust.ac.ir (A. Asad).

http://dx.doi.org/10.1016/j.micpro.2017.03.011
0141-9331/

the overall chip power budget in the nanoscale era, cache hierarchies have become substantial power consumers in future many-core CMPs.

A majority of prior researches on power management techniques in multicore processors focus on core designs to control the power consumption. The only knob that they use to manage the power of multicore systems is at the core level [35–48,69–73]. In this work, we show that uncore components such as cache hierarchy, on-chip interconnect and etc. are significant contributors in the overall chip power budget in the nanoscale era and play important roles in the dark silicon era. Uncore components, especially those in the cache hierarchy, are the dominant leakage consumers in multi/many-core CMPs. Therefore, besides focusing on energy-efficient core designs, how to design the uncore components is essential to tackle the challenges of multicore scaling in the dark silicon era.

Since the slight improvement in CMOS device's power density leads to the dark silicon phenomenon, the emerging power-saving materials manufactured with nano-technology might be useful for illuminating the dark area of future CMPs. The long switch delay and high switch energy of such emerging low-power materials are the main drawbacks which prevent manufactures from completely replacing the traditional CMOS in future processor manufacturing [8]. Therefore, architecting heterogeneous CMPs and integrating cores and cache hierarchy made up of different materials on the same die emerges as an attractive design option to alleviate the power constraint. In this work, we use emerging technologies, such as three-dimensional integrated circuits (3D ICs) [9,10] and non-volatile memories (NVMs) [11,12] to exploit the device heterogeneity and design of dark-silicon-aware multi/many-core systems.

With increasing parallelism levels of new applications (from emerging domains such as recognition, mining, synthesis and especially mobile applications) which can efficiently use 100–1000 cores, shifting to multi/many core designs has been aimed in recent years. Due to the scalability limitations and performance degradation problems in 2D CMPs, especially in future many-cores, in this work, we focus on 3D integration to reduce global wirelengths and improve performance of future CMPs. Among several benefits offered by 3D integrations compared to 2D technologies, mixed-technology stacking is especially attractive for stacking NVM on top of CMOS logics and designers can take full advantage of the attractive benefits that NVM provides.

In these days, providing analytical models for future multi/many-core CMPs in the presence of dark silicon is essential [4]. None of the previous studies have presented analytical models for the future CMPs. To the best of our knowledge, this is the first work which proposes an accurate power model that formulates the power consumption of future many-core CMPs.

With increasing core count and parallelization of applications, this issue has become important that in future many-core architectures, workloads are expected to be multithreaded applications. Most of power budgeting and performance optimization techniques proposed so far in the multicore systems [35–48,69–71] only focus on multiprogramed workloads where each thread is a separate application. These techniques are inappropriate for multithreaded applications. Our proposed analytical power model formulates the power consumption of CMPs with stacked cache layers under execution of both multiprogramed and multithreaded workloads, for the first time. Unlike the previous researches on dark silicon which consider only the portion of power consumption related to on-chip cores [2,13–24], the proposed model considers power impact of uncore components as the important contributors in the total CMP power consumption. Moreover, prior researches [69–71] as the latest works on performance/energy optimizing in multicore systems, do not support more than eight cores multicore. They are not scalable to many-core CMPs and they do not support non-uniform cache architecture (NUCA) as the main cache organizations for future many-cores. To the best of our knowledge, this is the first study that presents an accurate power model for many-core CMPs with stacked cache hierarchy. This analytical power model is useful for dark-silicon modeling in future. The proposed power model considers microarchitectural features and workload behavior. A recent study from industry [76] has revealed that under the same power budget, the best power allocation strategy depends strongly on application characteristics.

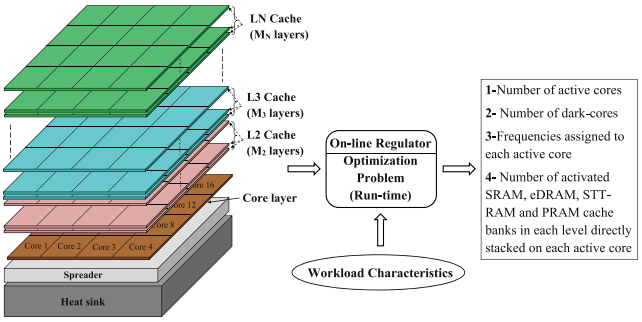


Fig. 1. Overview of the proposed architecture and the run-time flow.

In particular, in a dark-silicon-aware CMP with different components including cores and uncores, an integrated reconfiguration approach is needed at runtime to maximize performance under power and thermal constraints as well as under dynamic changing program behavior and execution parameters. To reach this target, based on the derived accurate power model for the proposed heterogeneous 3D CMP, two optimization problems have been formulated. These optimization problems are applied at run-time to reconfigure cache hierarchy and assign appropriate frequency/voltage to each core of the 3D CMP based on online workload characteristics monitoring. Since the proposed reconfiguration scheme should be aware of runtime application variability, such as program phase changes, it needs to be efficient enough to be brought at runtime.

Fig. 1 shows an overview of the proposed 3D CMP architecture with stacked heterogeneous cache hierarchy on the core layer and used run-time flow in this paper. As shown in this figure, the memory technology can be different between the levels in the cache hierarchy, while the memory technology is homogeneous within each level.

In continue, based on the prepared hardware required for the proposed reconfiguration approach, we propose a mapping technique for the target 3D CMP which considers the workload behavior to improve the thermal distribution at runtime, as an essential need in future many-core 3D CMPs.

The contribution of this paper is as follows:

- We propose an analytical power model that formulates the power consumption of future many-core CMPs. Specifically, this power model is useful for dark silicon modeling and can help to researchers to propose new power management techniques in future CMPs.
- We target CMPs with large number of cores (e.g., more than eight (many-core)) which require building a non-uniform cache architecture (NUCA) through a scalable network-on-chip (NoC) in order to reduce cache access latency, in this modeling for the first time.
- We consider the impact of power consumption of core and uncore components in parallel in this modeling for the first time.
- We consider both heterogeneous and non-heterogeneous CMPs under execution of both multiprogramed and multithreaded workloads in this modeling.

- We consider core and uncore leakage power consumption as an important contributor in the overall CMP power consumption in the nanoscale era in this power model.
- We propose an optimization-based power and thermal aware reconfiguration technique for the target dark silicon aware 3D CMP based on the derived power model.
- We consider microarchitectural features (core microarchitectures, cache organization, interconnection network and chip organization) and workload behavior (memory access pattern, dynamic changing program behavior and execution parameters) in the proposed reconfiguration technique.
- We propose a low-overhead mapping technique which considers the behavior of applications/threads at runtime based on the prepared hardware required for the proposed reconfiguration approach to balance the thermal distribution.

The rest of this paper is organized as follows. Section 2 analyzes the power consumption of cores and uncore components in CMPs. Section 3 reviews prior related works and background. Section 4 explains about the proposed heterogeneous 3D CMP architecture and the motivation behind of the proposed technique in this work. Section 5 presents the power model of the proposed 3D CMP with the stacked cache hierarchy. Section 6 describes the details of the optimization-based runtime reconfiguration technique which consists of two phases, online and off-line. Section 7 presents the proposed runtime application-aware mapping technique. Section 8 presents experimental results, and Section 9 concludes the paper.

2. Analyzing the contribution of cores and uncore components in total multicore processors power consumption

In this section, we analyze the power consumption of cores and uncore components in multicore systems. We first illustrate that uncore components have significant contribution in on-chip power consumption and we cannot ignore the impact of them in future chips' total power budget. We then show that the percentage of leakage power, a major fraction of total power consumption in uncore components, increases when compared to dynamic power as technology scales, and considerably outweighs the dynamic power in future nanoscale designs. To better understand the power distribution of a multicore processor, we use McPAT [60] and evaluate the power dissipation of cores and uncore components including L2/L3 cache levels, the routers and links of NoC, integrated memory controllers and integrated I/O controllers etc.

Fig. 2 illustrates the power breakdown of a multicore system with increasing number of cores under limited power budget. We use technology 32 nm in this figure. As shown in this figure, the power consumption of uncore components become more critical when the number of cores is increased in a multicore system and the power budget is a design constraint. In this work, we assume idle cores can be gated-off (dark silicon) while other on-chip resources stay active or idle under limited power budget. Actually, the uncore components remain active and consume power as long as there is an active core on the chip. As illustrated in Fig. 2, more than half of the power consumption is due to the uncore components in the 16-core and 32-core systems. Also, Fig. 2 shows that cache hierarchy and NoC consume a large portion of uncore power consumption. Therefore besides energy-efficient core designs, how to architect the uncore components is essential to tackle the challenges of multicore scaling in the dark silicon era.

As shown in Fig. 3 when technology scales from 32 nm to 22 nm, the ratio of leakage power increases and is expected to exceed the dynamic power in the future generations. We use 1 GHz frequency and 0.9 V supply voltage for an 8-core system in 32 nm and 22 nm technologies in Fig. 3. This figure shows that leakage

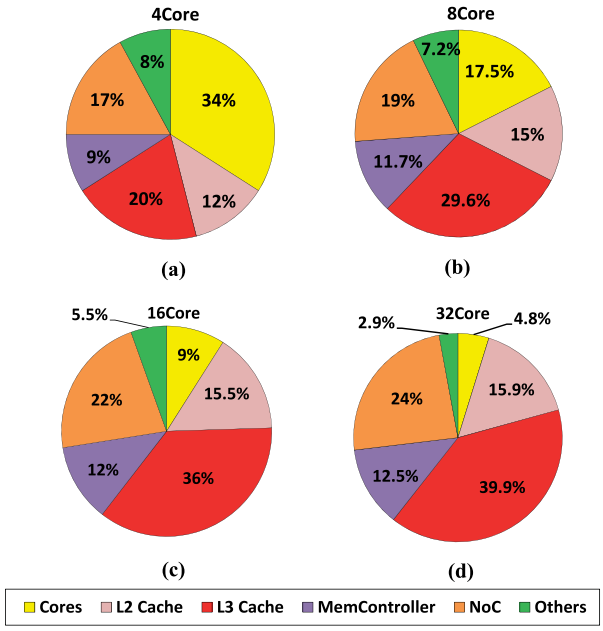


Fig. 2. Power breakdown of (a) a 4-core, (b) an 8-core, (c) a 16-core, and (d) a 32-core system under limited power budget.

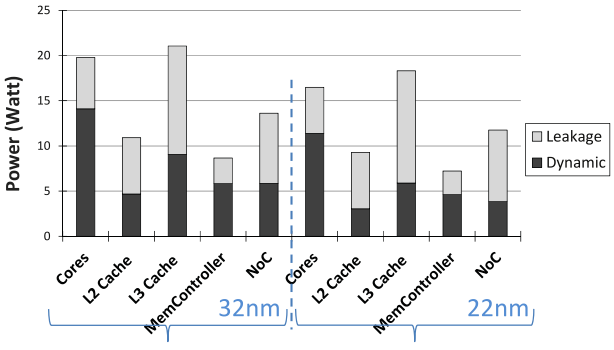


Fig. 3. Dynamic vs. leakage power for an 8-core system in 32 & 22 nm.

power dominates the power budget in the nanoscale technologies and is a major driver for unusable portion or dark silicon in future many-core CMPs. Thus, using emerging technologies such as NVMs with near-zero leakage power and three-dimensional integrated circuits (3D ICs) for stacking different technologies onto CMOS circuits bring new opportunities to the design of multi/many-core systems in the dark silicon era.

3. Background and related work

3.1. Background

Compared with traditional memory technologies such as SRAM and DRAM, NVM technologies commonly offer many desirable features like near-zero leakage power consumption due to their non-volatile property, high cell density and high resilient against soft errors. Nevertheless, they suffer from some obstacles such as limited number of write operations and long write operation latency and energy. Table 1 lists a brief comparison between SRAM, STT-RAM, eDRAM and PCRAM technologies in 32 nm technology. The estimation is given by NVSim [58], a performance, energy, and area model based on CACTI [59]. Table 1 shows that the STT-RAM technology is around four times denser than SRAM. In addition, as shown in Table 1, STT-RAM has a much smaller leakage power

Table 1
Different memory technologies comparison at 32 nm.

Technology	Area	Read latency	Write latency	Leakage power at 80°C	Read energy	Write energy
1MB SRAM	3.03 **mm**^{2}	0.702 **ns**	0.702 **ns**	444.6 **mW**	0.168 **nJ**	0.168 **nJ**
4MB eDRAM	3.31 **mm**^{2}	1.26 **ns**	1.26 **ns**	386.8 **mW**	0.142 **nJ**	0.142 **nJ**
4MB STTRAM	3.39 **mm**^{2}	0.880 **ns**	10.67 **ns**	190.5 **mW**	0.278 **nJ**	0.765 **nJ**
16MB PCRAM	3.47 **mm**^{2}	1.760 **ns**	43.74 **ns**	210.3 **mW**	0.446 **nJ**	0.705 **nJ**

than SRAM. Also, STT-RAM has significantly high write latency and write power consumption compared with SRAM.

3.2. Related work

A number of recent researches over the past five years have addressed the dark silicon phenomenon [2,13–24,66].

To combat dark silicon, Esmailzadeh et al. [2] focused on using only general-purpose cores. They evaluated homogeneous dark silicon CMPs and showed that fundamental performance limitations stem from the processor core. They ignored the power impact of "uncore" components such as the cache hierarchy, memory subsystem and on-chip interconnection. In their paper, it is described that with technology scaling and increasing number of cores on a chip in CMPs, the number of these "uncore" components will increase and hence they will further eat into the power budget, reducing speedups. Ignoring power impact of uncore components has been mentioned as one of the limitation of this work.

The research in [14–17] works on architectural synthesis of heterogeneous dark silicon CMPs from performance and reliability aspects under power/area constraints. Turakhia et al. [15] proposed a framework for architectural synthesis of dark-silicon CMPs. Raghunathan et al. [14] proposed a framework to evaluate the benefits of selecting the more suitable subset of cores for an application in a dark silicon multi-core system to maximize performance within the power budget. Similar to Esmailzadeh et al. [2], works in [14–17] proposed design-time solutions.

There have been recent efforts to mitigate the impact of dark silicon using device level heterogeneity for processing elements. For example, variable symmetric multiprocessing (vSMP) [20] is an energy-efficient methodology presented by NVIDIA where cores with the same architecture, but fabricated by a different silicon process, are integrated. Some of them are using special low power silicon process while some are using standard silicon process. In another effort in the same category, authors in [21] use a combination of steep-slope devices (e.g., interband tunnel field-effect transistors (TFETs)) and CMOS devices in the design of heterogeneous multicores. The main idea in these studies is to dynamically switch between the processors based on the system workload as they have different performance-energy consumption behaviors.

In [22,23], near-threshold computing is an approach which allows cores to operate at a supply voltage near the threshold voltage and allowing several otherwise dark cores to be turned on.

Venkatesh et al. in [24] introduce the concept of "conservation cores". They are specialized processors that focus on reducing energy instead of increasing performance, used for computations that cannot take advantage of hardware acceleration.

The work in [66] targets architectural synthesis of heterogeneous dark silicon processors from performance aspect under power constraint. In order to maximize performance in [66], cores that are homogeneous but synthesized with different power/performance targets are exploited.

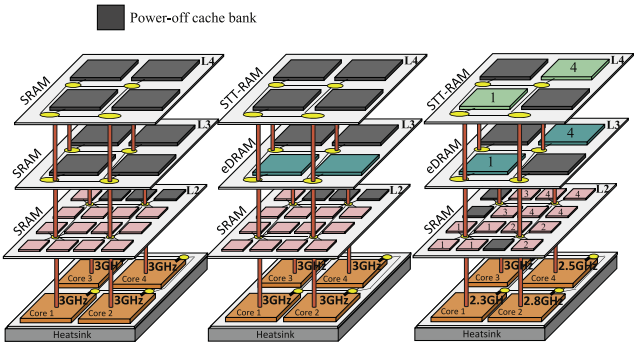
All of these prior works on the dark silicon [2,13–24,66] are characterization studies and focus on cores rather than uncore components. In one of the newest papers [73], authors review some recent papers in multicore systems which work on cores or uncore components separately. This paper advises to researchers to consider core power consumption in parallel with uncore components simultaneously for future many-core designs.

In this work, we consider the power impact of uncore components parallel with that of cores, for the first time.

To improve the performance of CMPs and reduce their power consumption, a number of researchers proposed 3D CMP architectures with 3D stacked memory/cache layers on top of a core layer [9,10,25–27,65,72,77]. In these studies, stacking large SRAM cache or DRAM memory on a core layer increased the performance and reduced the power consumption. Study in [10] demonstrated that up to sixteen layers of DRAM could be stacked on a quad-core processor without exceeding the maximum thermal limit. Cheng et al. in [77] proposed an energy efficient SRAM based last level cache without considering power and thermal limits. Stacked traditional memories such as SRAM or DRAM on the core layer may cause a drastic increase in power density and temperature-related problems such as negative bias temperature instability (NBTI) [28]. For example by stacking eDRAM/DRAM on top of cores as cache/main memory, the heat generated by the core-layer can significantly aggravate the refresh power of DRAM layers and the designer needs to consider the power consumption due to refresh when designing the power management policy for stacked DRAM memory or cache. Recently, emerging nonvolatile memory (NVM) technologies have emerged as candidates for future universal memory and cache subsystems due to their advantages such as high density, near-zero low leakage power, scalability and 3D integration with CMOS circuits [11,12,29–32]. Even though NVMs have many advantages as described above, shortcomings such as high write energy consumption, long latency writes and limited write endurance prevent them from being directly used as a replacement for traditional memories. To tackle these issues, recent studies [33–35] have proposed hybrid architectures, wherein traditional memories is integrated with NVMs to use advantages of both technologies. However, none of the aforementioned studies have explored these emerging memory technologies in the dark silicon context.

A number of researches proposed some proactive techniques to reduce the power consumption in multicore systems such as dynamic voltage and frequency scaling (DVFS) technique, thread scheduling, thread mapping, shutting-down schemes, and migration policies [36–43]. These aforementioned methods have not been designed for the dark silicon era. They cannot guarantee a situation that the system does not have enough power budget to keep running in the current setting. Also, these approaches limit their scope only to cores. In a power emergency situation, a well-designed power management method should not only decrease the power consumption to meet the new power constraint but also reduce the impact on performance as much as possible. In addition, prior work [36–42] provide power management for platforms implemented by using technology nodes in which dark silicon issue does not practically exists (e.g. 45 nm CMOS technology) and leakage power is not so problematic.

Although some efforts have been expanded in recent years, they are still relatively small improvement in mitigating the dark silicon issue due to dynamicity of the dark/dim area as it grows and shrinks at run-time. A broad open question that is unaddressed in the literature is the run-time optimization of the subset of cores to be kept dark and select the ideal set of on-chip resources to power



(a) Fixed core frequencies: 3GHz, SRAM Cache Levels. (b) Fixed core frequencies: 3GHz, Hybrid Cache Levels. (c) DVFS, Hybrid Cache Levels.

Fig. 4. The motivational examples of 3D CMPs with heterogeneous cache hierarchy.

on based on the available power budget, workload characteristics and the thermal profile of the chip. This motivates us to provide a comprehensive dark silicon aware runtime power and thermal management platform to react to power emergencies for future CMPs. Specifically, in this paper, we focus on power consumption of core and uncore components that interact with each other during applications execution time. Exploiting NVM technologies and 3D techniques in uncore components of the proposed CMP bring new opportunities to combat dark silicon challenge in this paper.

4. Proposed architecture

The architecture model assumed in this work is based on a 3D CMP with multi-level hybrid cache hierarchy stacked on the core layer similar to Fig. 1. As shown in this figure, each cache level is assumed to be implemented using a different memory technology. For motivating about the proposed architecture, we design two scenarios. In the first scenario, we consider a 3D CMP with homogenous cache hierarchy. In this scenario, we assume there is one layer per each level in the homogenous cache hierarchy stacked on the core layer such as Fig. 4(a). Also, we assume there are four cores in the core layer, each of them running *art* application [49]. Table 2 gives the properties of average memory access time (AMAT), as the performance parameter for evaluation of cache systems performance, and system power consumption when the stacked cache levels in the homogenous hierarchy are made from SRAM, eDRAM, STT-RAM, or PRAM.

Note that normalization reported in Table 2 is done based on the best case. That is power consumption is normalized with respect to the SRAM, whereas AMAT is normalized with respect to the PRAM. Based on these views, SRAM is fastest and higher power hungry option and it is better to use in lower level in the cache hierarchy because of faster accesses.

In this context, we introduce the steady-state temperature of the only layer of each cache level in the homogenous cache hierarchy shown in Fig. 4(a) as another measure to better understand. Fig. 5 depicts the temperature of the up layer of each cache level in the homogenous hierarchy shown in Fig. 4(a) vs. AMAT shown in Table 2. As shown in Fig. 4(a), since each cache level has one layer, the top layer is the single layer.

Table 2
Comparison of AMAT and system power consumption.

Technology	AMAT	Power consumption
SRAM	0.09	1
eDRAM	0.16	0.62
STT-RAM	0.3	0.37
PRAM	1	0.22

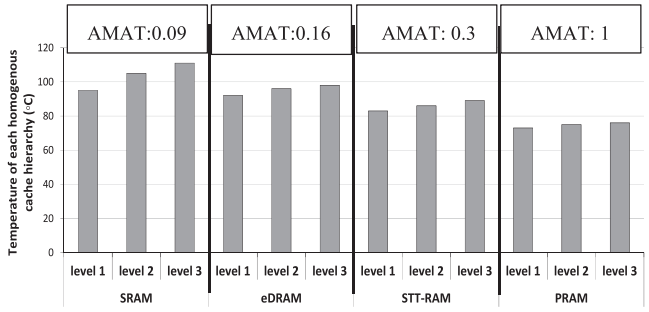


Fig. 5. Comparison of temperature of each homogenous cache hierarchy with respect to the AMAT shown in Table 2.

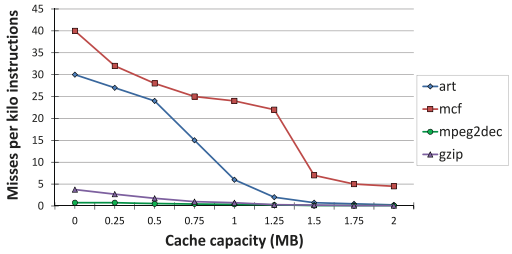


Fig. 6. Cache misses per instructions with respect to increasing capacity.

According to Fig. 5, if T_{max} is set to 80 °C, just the cache hierarchy based on PRAM satisfies the maximum temperature constraint, while it has the maximum AMAT in compared to the others. If T_{max} is set to 90 °C, STT-RAM and PRAM become suitable solutions, and STT-RAM is finally chosen to minimize AMAT. In some high-speed applications with T_{max} set to (90 °C ∼ 98 °C), the memory technology and the number of cache layers are selected between SRAM and eDRAM to minimize AMAT. If T_{max} is set to more than 100 °C, the best option to minimize AMAT is SRAM.

According to Table 2 and Fig. 5, amongst the homogeneous cache hierarchy, there exists a SRAM configuration which minimizes AMAT under the maximum temperature constant. Thus, there is no single memory technology in this study that has the best performance for all temperature ranges. This motivates us to study temperature-aware reconfigurable heterogeneous cache hierarchies, which combine the advantages of all these memory technologies to minimize power consumption and improve overall performance.

Based on the observations in Table 2 and Fig. 5, we decided to use SRAM in the L2 cache level, eDRAM in the L3 cache level, STT-RAM in the L4 cache level, and PRAM in the L5 cache level.

In the second scenario illustrated in Fig. 4, we consider three different implementations of our architecture with three cache levels in the hierarchy stacked on the core layer and other more details used in this paper. On the other hand, Fig. 4(c) illustrates an example of the proposed architecture shown in Fig. 1. We assume that *art, gzip, mpeg2dec*, and *mcf* [49] runs on Cores 1 to 4, respectively. In order to give more details about these four applications, Fig. 6 demonstrates the reduction of cache miss rate as the amount of cache assigned to programs increases. According to this figure, *mcf* and *art* are memory-intensive benchmarks since they show a largely reduction in cache miss rate as the cache capacity increases. Also, *mpeg2dec* and *gzip* are computation-intensive benchmarks because they show very small reduction in cache miss rate with increasing cache capacity.

In the second scenario, maximum temperature limit and power budget for the chip, T_{max} and P_{budget} are considered 80 °C and

Input	Output								
Core layer temperature	L_2 - SRAM	L_3 - SRAM	L_4 - SRAM	L_5 - eDRAM	f_1 (GHz)	f_2 (GHz)	f_3 (GHz)	f_4 (GHz)	IPS_{total} $(E+9)$
60 °C	2 layers	0	0	0	3 GHz	3 GHz	3 GHz	3 GHz	10.62
80 °C	1 layer	0	0	0	3 GHz	3 GHz	3 GHz	3 GHz	8.85

(a) Fixed core frequencies, L1, L2, L3: SRAM, L4: eDRAM.

Input	Output								
Core layer temperature	L_2 - SRAM	L_3 - eDRAM	L_4 - STTRAM	L_5 - PRAM	f_1 (GHz)	f_2 (GHz)	f_3 (GHz)	f_4 (GHz)	IPS_{total} $(E+9)$
60 °C	2 layers	2 layers	3 layers	1 layer	3 GHz	3 GHz	3 GHz	3 GHz	19.11
80 °C	1 layer	1 layer	2 layers	1 layer	3 GHz	3 GHz	3 GHz	3 GHz	15.93

(b) Fixed core frequencies, Heterogeneous cache hierarchy.

Input	Output								
Core layer temperature	L_2 - SRAM	L_3 - eDRAM	L_4 - STTRAM	L_5 - PRAM	f_1 (GHz)	f_2 (GHz)	f_3 (GHz)	f_4 (GHz)	IPS_{total} $(E+9)$
60 °C	2 layers	3 layers	3 layers	2 layers	2.76 GHz	2.10 GHz	2.61 GHz	2 GHz	26.75
80 °C	1 layer	3 layers	2 layers	1 layer	2.20 GHz	1.65 GHz	1.72 GHz	1.5 GHz	22.30

(c) DVFS, Heterogeneous cache hierarchy.

Fig. 7. An example for a 3D CMP with more than one layer in each level of the hybrid cache hierarchy.

100 *W*, respectively. Also, cache banks power-gating and per-core DVFS for the 3D CMP are assumed. Because of strong thermal correlations between a core and cache banks directly stacked on the core, the core and the cache banks in the same stack called a core-stack in our architecture.

In the three parts shown in Fig. 4, cache banks in each level of the hierarchy are allocated to cores such that IPS (instruction per second) is maximized without violating the maximum temperature limit and power budget. We also assume that the core clock frequency varies from 2 GHz to 3 GHz. In this motivational example, we assume one layer per each cache level. As shown in Fig. 4(a), the high leakage power consumption of SRAM technology has increased the temperature of layers of the cache hierarchy. To keep the temperature within the given limit, L3 and L4 levels are turned off. In Fig. 4(b), without violating the maximum temperature and power budget, in addition to L2, L3 cache banks are turned on because of lower leakage power of eDRAM based cache banks. In Fig. 4(c), more cache banks are allocated to each core in the upper level of the hierarchy by analytically determining the voltage/frequency of cores. In the Fig. 4(c), we allocate more cache banks from different technologies to the cores while lowering their frequencies and voltages in order to maximize the IPS and satisfy the temperature limit. Since the multiprogramed applications have high bandwidth demand, in Fig. 4(a) and (b), allocated cache banks to the cores are small according to the maximum temperature limit and power budget. Therefore, we can use allocated cache banks in each level in a shared manner which has its own problems. For example, there is contention between the working sets of different applications in the shared space. The proposed technique in Fig. 4(c) partitions the shared cache space according to the specific demands of each individual application in a workload set. Cache banks allocated to each core in each level are specified by the core numbers in Fig. 4(c). Increasing cache capacity yields significantly different performance improvement for various applications. This is due to the fact that some applications only need a small amount of cache while others benefit from larger caches and they use as much as available cache capacity given to them. Therefore, our proposed approach can be very useful because it reconfigures cache hierarchy based on the needs of the mapped application on each core. For example since mapped applications on Core1 and Core4 are memory-bound, more cache banks are allocated to Core1 and Core4 in Fig. 4(c).

In Fig. 7, we provide another CMP example with stacked cache hierarchy with more than one layer in each level. In this motivational example, similar to Fig. 4, we use four cores in the core layer of the 3D CMP. The applications mapped on cores in this example are *art, gzip, mpeg2dec*, and *mcf* running on Cores 1 to 4, respectively, same as previous example. The stacked cache hierarchy on the core layer of this 3D CMP is composed of L2 with two layers, L3 with three layers, L4 and L5 with four layers. T_{max} and P_{max} are same as Fig. 4. In Fig. 7, we report the number of allocated cache layers in each level to the whole of the core layer for simplicity. Also, we report the result for the core layer with two temperature, 60 °C and 80 °C. As shown in Fig. 7(a), L2, L3, and L4 are homogenous and made of SRAM and L5 is eDRAM. In Fig. 7(b) and (c), the stacked cache hierarchy is hybrid. These figures show that the frequency and voltage of cores depend on the number of active cache layers stacked directly on the core layer. As shown in Fig. 7(c), without violating the T_{max} and P_{max} by lowering frequency and voltage of some cores in the core layer, the number of active cache layers stacked directly on the cores can be increased to maximize the IPS. In Fig. 7, IPS result for each core layer's temperature has been normalized with respect to the Fig. 7(a). Details of the estimations and experimental setup used in this motivational example will be shown in Section 7.

By saving the leakage power from the heterogeneous cache hierarchy in the proposed architectural management technique in this work, the CMP would become more power-efficient and the saved power can be utilized to power-on darkened cores for performance improvement.

5. Power modeling for NoC-based many core CMPs

In this section, we present an analytical power model for future many-core chip multi-processors with multi-level cache hierarchy. The proposed model emphasizes on various types of on-chip resources such as cores, memory system and interconnection for the first time. The model can be very useful for future dark silicon aware power modeling in many core systems. Table 3 lists the parameters used in this model.

The total power consumption of a CMP mainly comes from three on-chip resources: cores, cache hierarchy, and interconnection network. Chip multiprocessors with a large number of cores (more than eight) require building architectures through a scalable network-on-chip (NoC). As widely used in the literature [13–24,35–48], we also adopt a mesh-based NoC in this modeling.

5.1. Components of the total power consumption of a 3D chip-multiprocessor

The total power of a 3D chip multi-processor can be calculated as the sum of the power of individual on-chip resources (cores and uncore components).

$$P_{Total} = P_{cores} + P_{uncores} \tag{1}$$

$$P_{Total} = P_{cores} + P_{cache\_hierarchy} + P_{interconnection} \tag{2}$$

5.1.1. Modeling core power consumption

We denote the power consumption of core i as P_i^{core}.

$$P_{cores} = \sum_{i=1}^{n} P_i^{core} \tag{3}$$

The power consumption of core i is comprised of dynamic and leakage power components. The total power consumption of core i is written as:

$$P_i^{core} = P_{D,i} + P_{L,i}\ ,\ \forall i \tag{4}$$

$$P_{D,i} = P_{max}\frac{f_i^2}{{f_{max}}^2},\ \forall i \tag{5}$$

Table 3
Parameters used in the power model.

Parameter	Description
n	Number of cores in the core layer
f_i	Operation frequency of core i
P_i^{core}	Power consumption of core i
$P_{D,i}$	Dynamic power consumption of core i
$P_{L,i}$	Leakage power consumption of core i
$P_i^{cache\_hirarchy}$	Sum of power consumption related to the dedicated cache banks in each level of the cache hierarchy to core i from the **1st** to the k**th** level
$P_{static_k}(T)$	Static power consumed by each layer of the k**th** cache level (L_k) at temperature TC
N	Number of cache levels ($L_1, L_2, \ldots, L_N$)
C_k	Capacity of the k**th** cache level (L_k)
$b_{i,k}$	Number of active cache layers in the region-set bank i stacked on core i at the k**th** cache level
$B_{i,k}$	Accumulated cache capacity in the region-set bank i stacked on core i at the k**th** cache level
$regn_k$	Total number of regions at the k**th** cache level (L_k)
a^r, a^w	Number of read and write accesses of an application
$APPH_k$	Average Power consumption Per Hit access
$x_{j,k}$	Indicator of j regions at the k**th** cache level
γ	Number of accesses per second
α	Sensitivity coefficient from the cache misses power-law
E_n	Data sharing factor of an application with n threads
T_s	Maximum execution time of the mapped applications
$E^s_{interconnection}$	Energy consumption of the interconnection network between nodes in T_s
$P_{interconnection}$	Power consumption of the interconnection between nodes
$P^q_{n,n',n''}$	Static power consumption of an interconnection network based on mesh topology with n nodes in dimension 1, n' nodes in dimension 2 and n'' nodes in dimension 3
E^s_{NP}	Average total energy dissipated in the on-chip interconnection network for transferring of NP packets in T_s
P^{qC}_R	Static power consumption of a router (without any packet)
P^c_R	Static power consumption of a router with one virtual channel (without any packet)
E^s_1	Average total energy dissipated for transferring of one packet from the source to the destination in the on chip interconnection network
E^P_R	Average energy dissipated in a router and the related link for a packet transfer
E^f_R	Average energy dissipated in a router and the related link for a flit transfer
D_{mesh}	The average distance of the mesh topology (The average number of links which a packet transits from the source to reach the destination)
v	Number of virtual channels per a link
l	Size of a packet based on number of flits

Since the operating voltage of a core depends on the operating frequency, it is assumed that the square of the voltage scales linearly with the frequency of operation [44]. In Eq. (5), P_{max} is maximum power budget and f_{max} is maximum frequency of the core.

The leakage power dissipation depends on temperature. The leakage power of core i can be written as Eq. (6). T_t is ambient temperature at time t and h_i is empirical coefficient for temperature-dependent leakage power dissipation. h_i coefficients in cores with same microarchitectures have the same value.

$$P_{L,i} = h_i \,.T_t, \quad \forall i, t \tag{6}$$

In this work for core power modeling, we can consider peak leakage power as other works [14,15]. Therefore, in this model we can use the maximum sustainable temperature for the chip.

$$P_{L,i} = h_i.T_{max}, \quad \forall i \tag{7}$$

5.1.2. Modeling cache hierarchy power consumption

a) Cache hierarchy power consumption modeling for multiprogramed workloads

As shown in Fig. 1, the number of cache levels is N and, each cache level is presented as L_k, $(k = 1, 2, 3, \ldots, N)$. In kth cache level, L_k, there is M_k layers and the lth cache layer in the $L_k$is represented as $A_{k,l}$ $(l = 1, 2, 3, \ldots, M_k)$.

We assume that in multi-programmed applications, each application mapped on each core effectively sees only its own slice of the dedicated cache banks in the cache hierarchy.

$$P_{cache\_hierarchy} = \sum_{i=1}^{n} P_i^{cache\_hierarchy} \tag{8}$$

$$P_i^{cache\_hierarchy} = P_{dynamic_i}^{cache\_hierarchy} + P_{static_i}^{cache\_hierarchy} \tag{9}$$

$$P_i^{cache\_hierarchy} = N_{access} \,.\, \sum_{k=1}^{N} m(C_{k-1}).h(C_k).E_{dyn_k}(C_k) + \sum_{k=1}^{N} P_{static}(T_t)_k \tag{10}$$

where h and m are hit and miss rates, respectively. Cache hit and miss rates depend on cache capacity. Increasing the cache capacity allocated to a core leads to reduce cache miss rate. E_{dyn_k} denotes dynamic energy consumed by kth cache level per access. N_{access} is number of accesses per second. $P_{static}(T_t)_k$ is static power consumed by kth cache level, L_k, with capacity C_k at temperature T_t.

The first part of Eq. (9), $P_{dynamic_i}^{cache\_hierarchy}$, depends on dynamic energy. Dynamic energy consumed by cache depends on Average Memory Access Time (AMAT). Reducing AMAT leads to lower cache dynamic energy.

For formulating first part of Eq. (10) based on *accessible variables* in the model, first we compute the average power per access (APPA) by:

$$APPA = APPH_1 + \sum_{k=1}^{N-1} APPH_{k+1} \,.\, R_k^{miss} \tag{11}$$

where R_k^{miss} is the product of cache miss rates from the 1st to the kth cache level. Since the access time of reading and writing in emerging non-volatile memories (i.e., STTRAM-based or PRAM-

based cache) is different, the $APPH_k$ is expressed as:

$$APPH_k = \frac{a^r \ . \ \tau_k^r \ . \ p_k^r + a^w \ . \ \tau_k^w \ . \ p_k^w}{a^r + a^w} \tag{12}$$

where a^r and a^w are the number of read and write accesses of the program running on a core. τ_k^r and τ_k^w are latencies of read and write at the kth cache level, and p_k^r and p_k^w are power consumption of read and write at the kth cache level, respectively. We can rewrite Eq. (12) as:

$$APPH_k = \frac{a^r . \ E_{read\,k} + a^w \ . \ E_{write\,k}}{a^r + a^w} \tag{13}$$

where $E_{read\,k}$ and $E_{write\,k}$ are read and write energy at the k*th* cache level, respectively.

$$R_k^{miss} = \mu \ . \left(\frac{B_k}{\sigma}\right)^{-\alpha} \tag{14}$$

where σ is baseline cache size. μ is baseline cache miss rate. α is power law exponent, typically lies between 0.3 and 0.7 [50]. B_k is the sum of allocated cache capacity from the 1st to the kth cache level and is obtained by:

$$B_k = \sum_{m=1}^{k} c_m.b_m \tag{15}$$

where c_m and b_m are the capacity of each cache layer and the number of active cache layers at the mth cache level, respectively.

We can rewrite the first part of the Eq. (10), $P_{dynamic_i}^{cache\_hierarchy}$, based on the *accessible variables* as:

$$P_{dynamic_i}^{cache\_hirarchy} = \gamma \ . \left(APPH_1 + \sum_{k=1}^{N-1} APPH_{k+1} \ . \ \mu \ . \left(\frac{B_{i,\ k}}{\sigma}\right)^{-\alpha}\right) \tag{16}$$

where γ is the number of accesses per second. In Eq. (17), d_i is the time-to-deadline constraint of the program allocated to core *i*.

$$N_{access} = \ \gamma = \frac{a^r + a^w}{d_i} \tag{17}$$

As one of the worst case, we can assume all of accesses of the mapped application are to the *N*th cache level of the hierarchy with biggest latency. Therefore, we can set d_i as:

$$d_i = \ a^r.\tau_N^r + a^w.\tau_N^w \tag{18}$$

The second part of Eq. (10), $P_{static_i}^{cache\_hierarchy}$, is the total leakage power consumption related to the dedicated cache banks to core i which is the main contributor to the total power consumption.

$$P_{static_i}^{cache\_hirarchy} = \sum\nolimits_{k=1}^{N} b_{i,\ k} \ .P_{static_k}(T_{max}) \tag{19}$$

$$b_{i,\ k} = \frac{B_{i,\ k} - B_{i,k-1}}{c_k} \tag{20}$$

$P_{static_k}(T_{max})$ is the static power consumed by each layer of the kth cache level (L_k) at temperature T_{max}.

Eqs. (10)–(20) model cache hierarchy power consumption in multi-programed workloads where each core runs a separate application.

$P_{i,\ k}^{cache}$ is used as the per-level power consumption in the region-set bank i stacked on core i at the kth cache level.

$$P_{i,\ k}^{cache} = P_{dynamic} + P_{static}, \forall i$$
$$= \left(APPH_k + APPH_{k+1}.\mu.\left(\frac{B_{i,\ k}}{\sigma}\right)^{-\alpha}\right) + b_{i,k}.P_{static_k}(T_{max}) \tag{21}$$

b) Cache hierarchy power consumption modeling for multi-threaded workloads

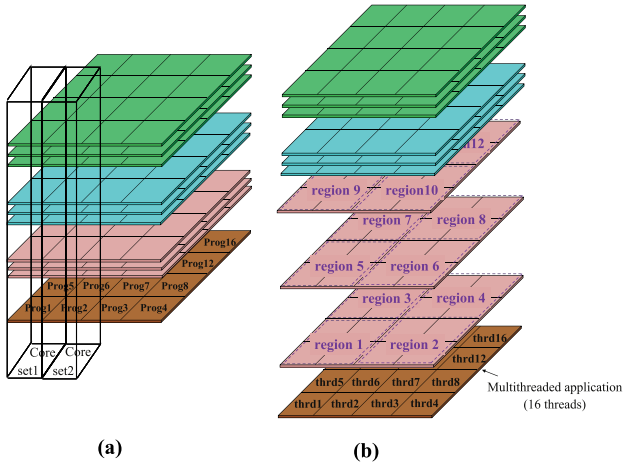


Fig. 8. The style of using cache hierarchy in: (a). A multiprogramed workload and, (b). A multithreaded workload.

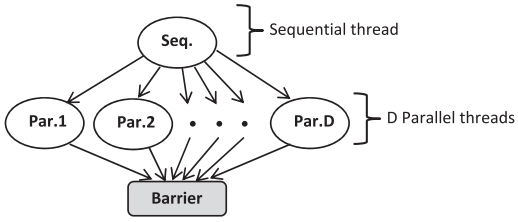


Fig. 9. An example of a multithreaded application with ***D*** threads.

In previous sub-section, we model cache power consumption in multiprogramed workloads which each program only uses the dedicated cache banks in its own core-set privately as shown in Fig. 8(a).

Large class of multithreaded applications are based on barrier synchronization and consist of two phases of execution (shown in Fig. 9): a sequential phase, which consists of a single thread of execution, and a parallel phase in which multiple threads process data in parallel. The parallel threads of execution in a parallel phase typically synchronize on a barrier. In the parallel phase, all threads must finish execution before the application can proceed to the next phase. In multithreaded workloads, cache levels are shared across the threads. In the parallel phase, threads share regions at each layer of the cache levels in the hierarchy as shown in Fig. 8(b). First, we dedicate region1 in each level to the threads. Then based on power budget and performance constraints in optimization techniques, we can increase the number of regions or keep it fixed in each level.

Since multithreaded applications use cache hierarchy in shared style, we can rewrite Eq. (9) for them as follows:

$$P_{cache\_hierarchy} = P_{dynamic}^{cache\_hierarchy} + P_{static}^{cache\_hierarchy} \tag{22}$$

We can rewrite Eq. (11) for a multithreaded program with more details as follows:

$$APPA = APPH_1 + \sum_{k\ =1}^{N-1} \sum_{j=1}^{regn_k} APPH_{j+1,k} \ .R_{j,k}^{miss} \tag{23}$$

Note that in Eq. (23), $APPH_{regn_k+1,k} = APPH_{1,k+1}$. It means after miss accessing to the last region of the kth cache level, search will be done in the first region of the next level. In this equation, $R_{j,k}^{miss}$ is the product of cache miss rates from 1st region of the 1st cache level to jth region of the k*th* cache level. The $APPH_{j,\ k}$ is average power per hit access to jth region of the kth cache level and ex-

pressed as:

$$APPH_{j,k} = \frac{a^r.\ \tau^r_{j,k}\ .\ p^r_{j,k} + a^w.\ \tau^w_{j,k}\ .\ p^w_{j,k}}{a^r + a^w} \tag{24}$$

Note that for all of the regions in each cache level, read latencies, write latencies, read energies and write energies are same, $\tau^r_{j,k} = \tau^r_{j+1,k}\ ,\ \tau^w_{j,k} = \tau^w_{j+1,k}\ ,\ p^r_{j,k} = p^r_{j+1,k},\ p^w_{j,k} = p^w_{j+1,k},\ \forall k, j$.

We can write Eq. (24) as:

$$APPH_{j,k} = \frac{a^r.\ E_{read\,j,k} + a^w\ .\ E_{write\,j,k}}{a^r + a^w} \tag{25}$$

In Eq. (23), R^{miss}_k for a multithreaded program can be modelled as:

$$R^{miss}_k = \mu\ .\ \left(\frac{B_k}{n\ .\ \sigma}\right)^{-\alpha}.\ E_n \tag{26}$$

where n is number of cores. σ is baseline cache size and μ is baseline cache miss rate.

$$R^{miss}_{j,k} = \mu\ .\ \left(\frac{B_{j,k}}{n\ .\ \sigma}\right)^{-\alpha}.\ E_n \tag{27}$$

$$B_{j,k} = \sum_{m=1}^{k}\sum_{j=1}^{regn_m} j.x_{j,k}.\frac{c_m}{regn_m} \tag{28}$$

$$\sum_{j=1}^{regn_k} x_{j,k} = 1,\ \forall k \tag{29}$$

Let $x_{j,k}, x_{j,k} \in \{0, 1\}, j \in [1, regn_k], k \in [1, N]$ be a binary variable. If it is 1, it shows that the multithreaded application uses region 0, region1, ..., region$j-1$ and region$j$ at the kth cache level. Note that $regn_k$ represents the total number of regions in kth cache level of the hierarchy. It is fixed for each cache level.

The first part of Eq. (22), $P^{cache\_hierarchy}_{dynamic}$, based on *accessible variables* is as follows:

$$P^{cache\_hirarchy}_{dynamic} = \gamma.\left(APPH_1 + \sum_{k=1}^{N-1}\sum_{j=1}^{regn_k} APPH_{j+1,k} .\mu\ .\ \left(\frac{B_{j,k}}{n\ .\ \sigma}\right)^{-\alpha}.\ E_n\right) \tag{30}$$

where γ is modeled as Eq. (17).

The second part of Eq. (22), $P^{cache\_hierarchy}_{static}$, is the total leakage power consumption related to the dedicated cache banks to core i which is the main contributor to the total power consumption.

$$P^{cache\_hirarchy}_{static} = \sum_{k=1}^{N}\sum_{j=1}^{regn_k} j.x_{j,k}\ .P_{static_k}(T_{max}) \tag{31}$$

where $P_{static_k}(T_{max})$ is the static power consumed by each region of the kth cache level, L_k, at temperature T_{max}. b_k, number of active cache layers in the kth cache level is $b_k = regn_k/4,\ k = 1, 2, \ldots, N$. In this model, we assume that each layer of cache levels in the hierarchy includes four regions.

5.1.3. Modeling on-chip interconnection power consumption

Energy consumption of the on-chip interconnection network in T_s is calculated as Eq. (32):

$$E^s_{interconnection} = E_{static} + E_{dynamic} = P^q_n\ .\ T_s + E^s_{NP} \tag{32}$$

$$E^s_{NP} = NP.E^s_1 = NP.(D_{mesh} + 1)\ .E^p_R = NP.(D_{mesh} + 1).l.E^f_R \tag{33}$$

In a mesh topology with d dimensions, which there is k_i nodes in ith dimension, the average distance that a packet must traverse to reach the destination can be calculated as Eq. (34):

$$D_{mesh} = \frac{1}{3}.\sum_{i=1}^{d}\left(k_i - \frac{1}{k_i}\right) \tag{34}$$

In a 2D mesh with $n$nodesin each dimension, the average distance between two nodes can be calculated as follows:

$$D_{mesh} = \frac{2n}{3} - \frac{2}{3n} \tag{35}$$

In a many-core platform based on 2D mesh topology ($n \geq 32$), the average distance is:

$$D_{mesh} \cong \frac{2n}{3} \tag{36}$$

Finally, power consumption of on-chip interconnection between nodes can be calculated as:

$$\begin{aligned} P_{interconnection} &= \tfrac{E^s_{interconnection}}{T_s} = P^q_{n,n',n''} + \tfrac{E^s_{NP}}{T_s} \\ &= n.n'.n''.P^{qC}_R + \tfrac{E^s_{NP}}{T_s} = n.n'.n''.\nu.P^c_R + \tfrac{E^s_{NP}}{T_s} \end{aligned} \tag{37}$$

where $T_s = \max(d_i),\ i = 1, 2, \ldots, n$, in Eq. (18).

Since Eq. (37) is the function of maximum execution time of the mapped applications, T_s, and T_s has a big value compare to E_{NP}, the second term of Eq. (37) can be ignored, therefore,

$$P_{interconnection} = n\ .\ n'.\ n''.\nu\ .P^c_R \tag{38}$$

As described in [55], also as shown in Fig. 3, particularly problematic for NoC structures is leakage power, which is dissipated regardless of communication activity. At high network utilization, static power may comprise more than 75% of the total NoC power at the 22 nm technology and this percentage is expected to increase in future technology generations. This fact is captured by Eq. (38).

6. Proposed power and temperature aware reconfigurable technique

6.1. High level system description

Power and thermal management techniques are receiving a lot of attention in future many-core systems. The proposed technique that dynamically adapts to the system is based on an optimization problem. We model this optimization problem based on the power model presented in Section 5. The optimization problem predicts the future thermal state of the system, improves the performance and minimizes power consumption by completely satisfying thermal constraints. Fig. 10 shows the block diagram of the proposed technique.

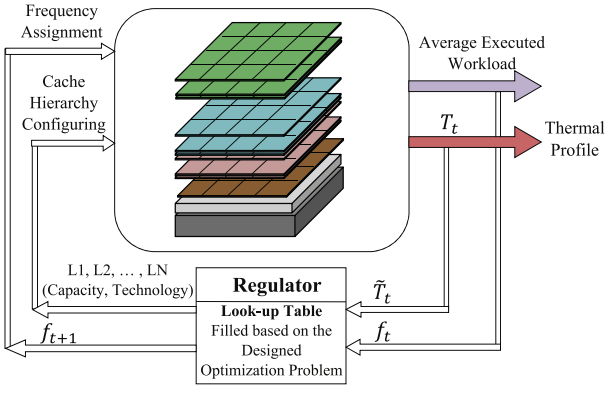


Fig. 10. Proposed temperature aware reconfiguration mechanism.

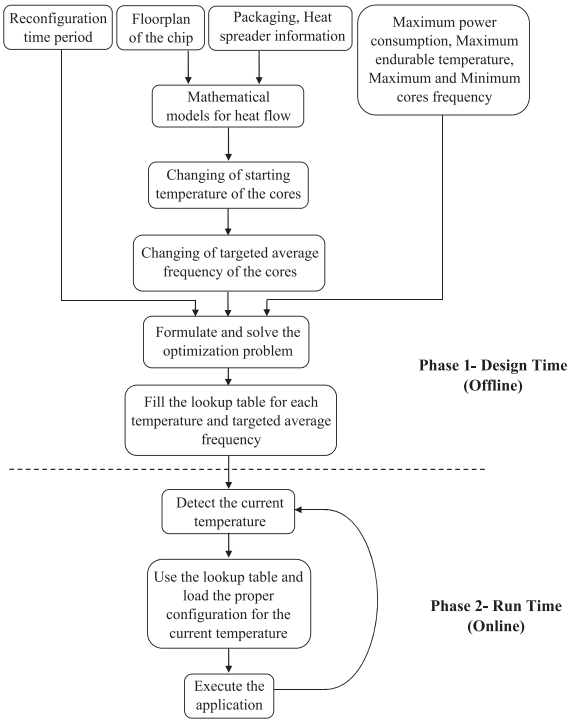


Fig. 11. Two Phases (online and offline) of the proposed scheme.

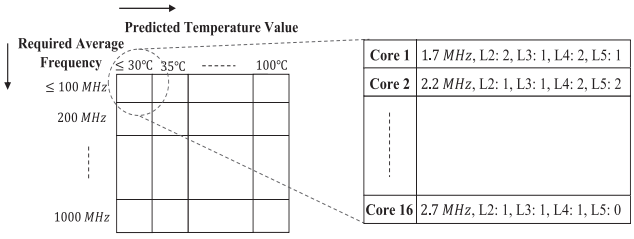


Fig. 12. A look up table provided at Phase 1.

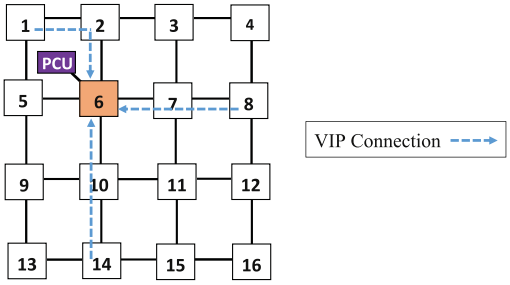


Fig. 13. Collecting information by the central monitoring tile 6 in the core layer and the location of the PCU.

The regulator which is based on the mentioned optimization problem monitors the 3D CMP state determined by temperature and working frequencies and makes decision about the configuration of the cache hierarchy and voltage/frequency for the system in the next time interval. Temperature is monitored by on-die thermal sensors. The proposed technique contains two phases, a design time phase (Phase 1) and a runtime phase (Phase 2). Designing the regulator based on the solving the optimization problem is done at design time in Phase 1. Monitoring the system status and making decision about the cache configuration and cores voltage/frequency in each interval are done at runtime in Phase 2. In next subsection, we explain about the proposed method operation in detail.

6.2. Proposed system operation

System operation can be divided into two phases: an off-line design time phase and run-time control phase. The overview of the aforementioned two phases is presented in Fig. 11.

6.2.1. Phase 1: design time

The inputs of this phase are the maximum power budget and the floorplan of the chip, the maximum and minimum operating frequencies of the cores, the time period at which the DVFS and cache hierarchy reconfiguration need to be applied, and the thermal models that are obtained based on the packaging and the heat spreader as shown in Fig. 11.

Fig. 12 shows a sample of look up table which is the output of phase1. In this table, for each starting temperature value and required average frequency of the running workload, a frequency vector and a cache hierarchy configuration (the capacity and technology of each level) are computed by solving an optimization problem described in Section 7.

6.2.2. Phase 2: run time

In typical CMP designs [51] and future many core systems, one or more Power Control Units (PCUs) are embedded. The PCU can be a dedicated small processor for chip power management as in Intel's Nehalem architecture or a specific low overhead hardware [51].

In the proposed architecture, we assume a single monitor tile that collects the statistical data from the whole network. Furthermore, in order to reduce the interconnection overhead between the PCU and monitor unit, monitor tile should be placed near the PCU. This proposed scheme is scalable to CMPs with large number of cores with more than one PCU.

In the proposed architecture, the PCU is in charge of conducting the reconfiguration process in a centralized manner in the on-line phase. The PCU utilizes the table obtained in off-line phase to set the frequencies of the cores and also, assign the capacity and technology of each level of cache in the hierarchy to each core (reconfiguration process). The reconfiguration procedure is applied periodically, at a pre-defined time period. A sample core layer which indicates tile 6 as the location of the monitor tile is illustrated in Fig. 13. Tile 6 is chosen as the location of the monitoring tile in this core layer since it is near to the other nodes to collect the desired statistics. Once the statistical data from the whole network arrive at the monitor tile, they are sent to the PCU. The PCU has adequate storage to accommodate the statistical data gathered.

It is assumed that each core has been equipped with a temperature sensor to report the current temperature of the core to the central monitoring tile in the proposed architecture. It should be noted that many of today's platforms (e.g. Versatile Express Development Platform [53] which includes ARM big.LITTLE chip) have been equipped with sensors to measure frequency, voltage, temperature, power, and energy consumption of each core or cluster [52]. Xu et al. [45] allocated a type of TSV for thermal monitoring for transferring temperature information of temperature sensors in addition to the data and control TSVs for the future 3D NoCs. Similarly, Bakker et al. [46] presented a power measurement technique based on power/thermal monitoring using on core sensors for Intel SCC [54].

At the end of each time interval, each core running a program monitors its current temperature and sends a control flit containing the temperature value to the monitoring tile. The control flit is shown in Fig. 14.

The communication between the monitor tile and other cores in the core layer is handled by virtual point-to-point (VIP) con-

FT	VC	MSG	Route Info.	Temperature Val.	Unused
2bits	2bits	8bits	8bits	8bits	100bits

Fig. 14. Fields of a control flit containing temperature information.

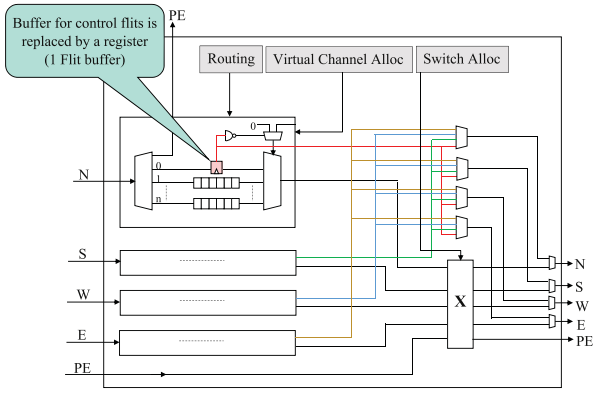


Fig. 15. Architecture of a router preparing VIP connections [47].

i	Frequency	Voltage	P/S	Number of Layers L2	Number of Banks L2	Number of Layers L3	Number of Banks L3	Number of Layers L4	Number of Banks L4	Number of Layers L5	Number of Banks L5	Unused
1bit	10bits	10bits	1bit	5bits	10bits	5bits	10bits	5bits	10bits	5bits	10bits	26bits

Fig. 16. Fields of a reconfiguration command flit sent by monitoring tile to each core.

nections as a separate control network. Architecture of the used routers preparing VIP connections in this work is shown in Fig. 15.

The VIPs between the monitor tile and other cores are constructed on demand at run-time over the virtual channels. VIPs bypass the intermediate routers, and VIP connections are constructed by borrowing one of the packet-switched virtual channels on top of a packet-switched network. As shown in Fig. 15, a register with the capacity of one flit replaces the regular buffers in one virtual channel (e.g. virtual channel 0) in each physical channel of the routers in a VIP-enabled NoC. The flit in the VIP register (virtual channel 0) is prioritized over regular VCs and is directed to the crossbar input when the register has incoming flits to service. Otherwise, a virtual channel is selected based on the outcome of the routing function like traditional packet-switched networks. VIP connections are not allowed to share the same links. At most one VIP connection can be used per each router port. In contrast to the dedicated point-to-point links which are physically established between the communicating cores and are fixed during the system life-time, VIP connections are dynamically reconfigurable and can be established based on the workload traffic pattern on the system. Modarressi et al. [47] and Asad et al. [48] provide additional details about VIP connections. The entire proposed reconfiguration procedure (making decision about cache hierarchy architecture and V/F of each core at each interval) is done very fast using VIPs and does not degrade the system performance considerably. The reconfiguration commands are sent over the VIPs to the cores by the monitor tile.

As shown in Fig. 16, the command flit specifies the voltage/frequency of core i and the number of activated layers and banks in each level of the cache hierarchy in shared or private use based on P/S bit. If this bit is 1, the cache levels are in the private use (multiprogramed), and if it is 0, the cache levels are in the shared use (multithreaded).

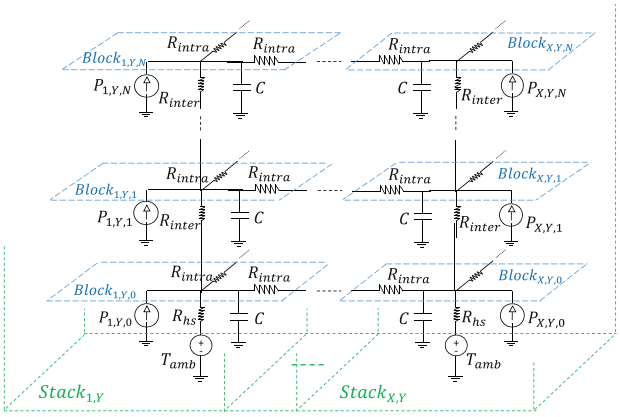


Fig. 17. Thermal model of a 3D IC system.

In the beginning of the each time interval, based on the gathered cores temperature information by the monitor tile, the PCU finds the maximum temperature across the cores and sets it as starting temperature value, T_{start}.

Also, based on the power budget,P_{budget}, the average frequency, $f_{avg}(T)$, and the amount of dissipated power consumption by cores and cache hierarchy, $P_{current}$, in the current interval, the PCU calculates the required average operating frequency across all cores for the next interval.

$$f_{avg}(T+1) = f_{avg}(T) + K.\left(P_{budget} - P_{current}\right) \tag{39}$$

In Eq. (39), $f_{avg}(T)$ is the frequency at Tth time interval, $f_{avg}(T+1)$ is the frequency at the time interval after Tth interval, K is a constant value computed based on the system power behavior, P_{budget} is the specified power budget and $P_{current}$ is the current power.

According to the T_{start} and predicted required average frequency of the cores for the next time interval, the PCU chooses the frequency assignment, and configure cache hierarchy for the cores from the filled look up table in phase 1. If the average frequency point cannot be supported in the table, the PCU chooses the next lower frequency row in the look up table.

6.3. Problem formulation

In this section, we investigate the details of optimization-based reconfiguration approach and formulate its fundamental goals during run-time. We use the power model obtained in Section 5 to formulate the optimization problem used by reconfiguration regulator. As mentioned before, future dark silicon CMPs consist of many cores where only few of them can be simultaneously powered on or utilized within the peak power and temperature budget. Assume that, peak power budget and temperature limit are given by a designer specified value, P_{budget}, and T_{max}.

Unlike much of the prior work [2,13–24], we consider temperature as a fundamental constraint in the dark silicon estimations. Dark silicon modeling under TDP constraint may lead either to underestimation or overestimation of dark silicon [2,66]. To provide more accurate analysis of dark silicon, temperature needs to be considered in estimating dark silicon. Therefore, we present a coarse-grained thermal model of a 3D CMP in this section.

6.3.1. Heat propagation model

In the thermal model of a 3D CMP used in this design, each block (e.g., core and cache bank) is represented by a set of thermal model elements (i.e., thermal resistance, heat capacitance, and current source) [63]. In Fig. 17, the heat sink is located at the bottom

of the chip stack and there are thermal resistances between horizontally adjacent blocks, R_{intra}. Also, there are thermal resistances between vertically adjacent blocks, R_{inter}. The power consumption of a block influences its temperature as well as the temperature of other blocks.

The steady-state temperature of each block (e.g., core and cache bank) can be calculated using this thermal model. For example, we can calculate the steady-state temperature of the thermal elements (*X*, *Y*, 0) and (*X*, *Y*, 1) in Fig. 17 as follows:

$$T^{ss}_{(X,Y,1)} = P_{(X,Y,1)} \ .R_{inter} + T^{ss}_{(X,Y,0)} \tag{40}$$

$$T^{ss}_{(X,Y,1)} = \left(P_{(X,Y,0)} + P_{(X,Y,1)}\right).R_{hs} + T_{start} \tag{41}$$

where $T^{ss}_{(X,Y,0)}$ and $T^{ss}_{(X,Y,1)}$ are the steady-state temperatures, and $P_{(X,Y,0)}$ and $P_{(X,Y,1)}$ are the power consumption of the blocks (*X*, *Y*, 0) and (*X*, *Y*, 1), respectively. T_{start} denotes the ambient temperature, and R_{hs} is the thermal resistance from thermal element (*X*, *Y*, 0) to the ambient through the cooling structure. In the target CMP, we call the core and the cache banks in the same stack as core-stack. Therefore based on Eqs. (40) and (41), the steady-state temperature of core *i* and cache banks stacked directly on it, *stack i*, can be obtained as follows:

$$T^{top}_i = \ T_{start} + \ R_{hs} \ .P^{core}_i + \sum_{k=1}^{N} R_{inter} \ . \ P^{cache}_{i,\ k} . \ b_{i,\ k}, \ \forall i \tag{42}$$

6.3.2. Objective functions and constraints

a) Optimization problem based on the multiprogramed workloads:

We will now propose an optimization strategy to determine the optimal heterogeneous dark silicon 3D CMP architecture for multiprogramed workloads. The outputs of the optimization problem are: 1) determining the optimal number of active cores, 2) finding which cores are turned on and which cores are left dark, 3) assigning frequencies (and the corresponding voltages) to each core, and 4) allocating the optimal number of SRAM cache banks in L2, eDRAM cache banks in L3, STT-RAM cache banks in L4, PRAM cache banks in L5 to each core and turning off unassigned cache banks in the hierarchy. The goal of the proposed optimization is to minimize power consumption under temperature limit and performance constraint.

In our model, optimal frequency of each core, $f_{cores} = \{f_1, f_2, \ldots, f_n\}$, and the optimal number of activated SRAM, eDRAM, STT-RAM and PRAM cache banks in each level directly stacked on core *i*, $L_i = \{L_1, L_2, \ldots, L_N\}$, are the optimization variables. The power optimization problem *J*1 is presented below:

$$\begin{aligned} minimize\ J1 = \ & \sum_{i=1}^{n} (P^{core}_i)_t + \sum_{i=1}^{n} \left(P^{cache\_hirarchy}_i\right)_t \\ & + P_{interconnection}, \ \forall t \end{aligned} \tag{43}$$

$$subject\ to: \ f_{min} \ \le (f_i)_t \ \le \ f_{max}, \ \forall t, \forall i \tag{44}$$

$$P_{max} \ .\left((f_i)^{\alpha}_t / f^{\alpha}_{max}\right) \ + \ h_i . \ (T^{core}_i)_{init} \le \ (P^{core}_i)_t, \ \forall t, \forall i \tag{45}$$

$$(T^{core}_i)_t = T_{start} + \ R_{hs} \ . \ (P^{core}_i)_t, \ \forall t, \ \forall i \tag{46}$$

$$(T^{top}_i)_t = \ (T^{core}_i)_t + \sum_{k=1}^{N} R_{inter} \ . \ (P^{cache}_{i,\ k})_t \ . \ b_{i,\ k} \ < T_{max}, \ \forall t, \forall i \tag{47}$$

$$\begin{aligned} (T^{core}_i)_{t+1} = \ & (T^{core}_i)_t \ + \sum_{\forall j \ \in \ Adj_i} a_{i,j}\left((T^{core}_j)_t - \ (T^{core}_i)_t\right) \\ & + R_{hs} \ . \ (P^{core}_i)_t < T_{max}, \ \forall t, \ \forall i \end{aligned} \tag{48}$$

$$\begin{aligned} \left(P^{cache\_hirarchy}_i\right)_t = \ & \gamma . \left(APPH_1 + \sum_{k=1}^{N-1} APPH_{k+1} . \mu . \left(\frac{B_{i,k}}{\sigma}\right)^{-\alpha}\right) \\ & + \sum_{k=1}^{N} b_{i,\ k} . \left(P_{static}\left(\left(T^{core}_i\right)_{int}\right)\right)_k, \ \forall t, \forall i, \forall k \end{aligned} \tag{49}$$

$$\sum_{i=1}^{n} (f_i)_t \ \ge n \times f_{avg}, \ \forall t, \forall i \tag{50}$$

$$b_{i,k} = \frac{B_{i,k} - B_{i,k-1}}{c_k}, \ \forall i, \forall k \tag{51}$$

$$P_{interconnection} = n.N.v.P^{c}_{R} \tag{52}$$

We assign $(T^{core}_i)_{init} = T_{start}$ for the $t = 1$, as the first time frame of each interval. If the scheme is applied every 100 ms and each time frame is 0.5 ms, then the total number of time frames are 200, $1 \le t \le 200$.

In Eqs. (45) and (49), $(T^{core}_i)_{init}$ is the temperature of the core layer which obtained by solving the optimization problem *J*1 time frame by time frame, $(T^{core}_i)_{init} = (T^{core}_i)_{t-1}$.

$P_{static}((T^{core}_i)_{init})$ is a constant value since we record the static power in different important temperatures in a table in the off-line phase.

Eq. (44) describes that working frequencies of cores are assumed continuous, ranging from f_{min} to f_{max}. Note that if the temperature of a core in a core-stack exceeds the critical point and does not decrease by reducing its frequency to the minimum level, then the PCU turns the core off at the beginning of the next time interval. In this case, to save power consumption and remain under T_{max}, the PCU groups low instruction per cycle (IPC) applications that may be running on two or more cores into one core. IPC as an appropriate performance parameter can be obtained from hardware performance counters which provided in most modern processors [51].

The heterogeneity in the operating frequency assigned to cores, and the heterogeneity in capacity and technology of each cache level in the hierarchy dedicated to each core in a CMP used in this optimization-based technique leads to heterogeneous CMPs which are envisioned to be a promising design paradigm to combat today's dark-silicon challenge.

The proposed optimization model prevents the unpredictable temperature variation in each time interval during runtime. Temperature variation causes extensive temperature-related problems in reliability especially in nano-scale designs. In this proposed model, Eq. (48) strongly prevents from operating temperature variation. Eqs. (45), (49) and (52), address the importance of the leakage power as an important factor in total power consumption of nanometer designs. More specifically, in the Eqs. (45) and (49), at the start of each time interval, leakage power has been calculated based on the starting temperature of the core layer.

In Eq. (50), f_{avg} is predicted based on Eq. (39) in each time interval. In Eq. (52), the on-chip interconnection power consumption is modeled as a function of the number of cache levels in the hierarchy. This is the first work which consider on-chip interconnection power in parallel with cache hierarchy, analytically.

NoC-Sprinting [67] is one of the newest topics in designing power-efficient NoCs in the dark silicon era. The proposed sprinting technique in [67] can activate a subset of network components

(routers and links) to connect a certain number of cores during workloads execution. Since the focus of [67] is on how to design interconnect, it does not consider any problems related to execution and cache systems. An efficient sprinting mechanism should be able to provide different levels of parallelism desired by different applications. Also, depending on workload characteristics, the optimal amount of cache and optimal number of cores is required to provide maximal performance speedup. Based on our proposed optimization model, we can turn on network components in the regions related to the turned-on cache levels and turn off others in the gated-off regions. Also, our model finds the optimal amount of cache and optimal number of cores. This proposed framework, as the first work, can help to researchers want to consider sprinting in their proposed techniques and models.

An alternate formulation of the energy efficiency optimization problem is to maximize performance under a fixed power budget. Here, instead of letting large parts of the multicore remain unused because of dark silicon, we adopt a dim silicon approach. In this approach, there are more cores sharing the available power budget and to find an energy-efficient runtime configuration, we exploit the applications' characteristics when distributing resources. Optimization variables of the optimization problem in Eqs. (53)–(62) are same as the above-mentioned optimization problem, Eqs. (43)–(52). Since in the later version power budget has been fixed by the designer at first, it is very useful for future dark-silicon aware CMP designing. This approach is presented as below:

$$maximize\ J2 = \sum_{i=1}^{n} (f_i)_t\ ,\ \forall t \tag{53}$$

$$subject\ to:\ f_{min}\ \le (f_i)_t\ \le\ f_{max},\ \forall t, \forall i \tag{54}$$

$$\sum_{i=1}^{n} (P_i^{core})_t + \sum_{i=1}^{n} \left(P_i^{cache\_hirarchy}\right)_t + P_{interconnection}\ \le\ P_{budget},\ \forall t \tag{55}$$

$$P_{max}\ .\left((f_i)_t^{\alpha} / f_{max}^{\alpha}\right)\ +\ h_i.\ (T_i^{core})_{init}\ \le\ (P_i^{core})_t,\ \forall t, \forall i \tag{56}$$

$$(T_i^{core})_t = T_{start} + R_{hs}\ .\ (P_i^{core})_t,\ \forall t,\ \forall i \tag{57}$$

$$(T_i^{top})_t = (T_i^{core})_t + \sum_{k=1}^{N} R_{inter}\ .\ (P_{i,\ k}^{cache})_t\ .\ b_{i,\ k} < T_{max},\ \forall t, \forall i \tag{58}$$

$$(T_i^{core})_{t+1} = (T_i^{core})_t + \sum_{\forall j\ \in\ Adj_i} a_{i,j}\left((T_j^{core})_t - (T_i^{core})_t\right) + R_{hs}.(P_i^{core})_t < T_{max}, \forall t, \forall i \tag{59}$$

$$\left(P_i^{cache\_hirarchy}\right)_t = \gamma.\left(APPA_1 + \sum_{K=1}^{N-1} APPA_{K+1}.\mu.\left(\frac{B_{i,k}}{\sigma}\right)^{-\alpha}\right) + \sum_{k=1}^{N} b_{i,\ k}.\left(P_{static}\left(\left(T_i^{core}\right)_{init}\right)\right)_k, \forall t, \forall i, \forall k \tag{60}$$

$$b_{i,k} = \frac{B_{i,k} - B_{i,k-1}}{c_k},\ \forall i, \forall k \tag{61}$$

$$P_{interconnection} = n.N.v.P_R^c \tag{62}$$

Eq. (55) represents the dark silicon constraint. This equation shows that peak power dissipation during the applications running must be less than the maximum power budget, P_{budget}.

b) Optimization problem based on the Multithreaded Workloads:

For applying of these two optimization problems for multithreaded applications, we should replace the term $\sum_{i=1}^{n} (P_i^{cache\_hirarchy})_t$ in Eqs. (49) and (60) by the presented term $P^{cache\_hierarchy}$, in Eq. (63), due to the shard style use of cache levels in the hierarchy by the multithreaded applications.

Therefore, Eqs. (49) and (60) are replaced with Eq. (63).

$$P^{cache\_hierarchy} = \gamma\left(APPH_1 + \sum_{k=1}^{N-1}\sum_{j=1}^{regn_k}\left(APPH_{j+1,k}.\mu.\left(\frac{B_{j,k}}{D.\sigma}\right)^{-\alpha}.E_D\right)\right) + \sum_{k=1}^{N}\sum_{j=1}^{regn_k} j.x_{j,k}.P_{static_k}(T_{max}) \tag{63}$$

where D is the degree of parallelism (DOP) of a multithreaded application (Fig. 9). We consider multithreaded applications with fix number of threads in this work. In the future work, we are going to apply this model for multithreaded applications with the variable DOP.

For thermal modeling, Eqs. (47) and (58) are replaced with Eq. (64).

$$\left(T_i^{top}\right)_t = \left(T_i^{core}\right)_t + \sum_{k=1}^{N} R_{inter}\ .\ \frac{P^{cache\_hierarchy}}{n}\ \le T_{max},\ \forall i, t \tag{64}$$

where n is number of cores in the core layer or number of tiles (spots) at the kth, $(k = 1,\ 2,\ \ldots,\ N)$, cache level.

In addition, Eq. (65) is added to two optimization problems for multithreaded applications.

$$\sum_{j=1}^{regn_k} x_{j,k} = 1,\ \forall k \tag{65}$$

Eq. (65) identifies the number of active regions in each cache level. $x_{j,\ k}$ is the optimization variable which shows the number of active regions in each level. Other equations in the presented optimization problems are same for the multithreaded version. In the multithreaded version, finding optimal frequency of each core, $f_{cores} = \{f_1, f_2, \ldots, f_n\}$, and the number of activated SRAM, eDRAM, STT-RAM and PRAM cache regions in each level directly stacked on the core layer, $regn_{caches} = \{x_{j,1}, x_{j,2}, \ldots, x_{j,N}\}$, are as optimization variables.

6.3.3. Architectural overhead

The regulator (shown in Fig. 10) needs an amount of hardware support. As described earlier in Fig. 11, the reconfiguration time period at which the proposed reconfiguration policy needs to be applied is obtained as an input. Therefore at each tile, one counter is needed to frame the control interval. In our case, we assume 100 ms as the maximum predetermined time interval by the designer, so 30 bits for the counter at each tile is sufficient. We assume the PCU is co-located with the monitor tile. Since the PCU already exists in typical CMP designs [51], all computations in our approach can be performed by the PCU without any extra hardware overhead.

The search operation for finding the frequency of the cores and configuration of the cache hierarchy for the next interval in the look up table based on temperature and average frequency is composed of a few simple calculations, which can be easily handled by the PCU. Therefore, the overall hardware overhead is 30 bits per tile.

6.3.4. Solving the proposed optimization problems

In the optimization problem presented in Eqs. (43)–(52) and Eqs. (53)–(62), the objective functions and constraints, except Eqs. (49) and (60), are linear. As discussed in [78], linear functions are convex. Eqs. (49) and (60) seem to be non-linear. Because of the fact that for convexity proof [78], all the constraints in the optimization problem should be convex functions, we show that Eqs. (49) and (60) are convex.

In Eqs. (49) and (60), the only term which we need to prove its convexity is:

$$g(c) = \left(\frac{B_{i,k}}{\sigma}\right)^{-\alpha} \tag{66}$$

Note that x^{β} is convex on R_{++} when $\beta \geq 1$ or $\beta \leq 0$ [78]. Based on this point, Eqs. (49) and (60) which include of summation and product of convex functions are convex. Therefore, objective function and all the constraints in the proposed optimization problems are convex. In this context, we can proof that Eq. (63) is convex.

To solve the optimization models, we use Maple [57], an efficient optimization solver. As the optimization models are solved for each temperature and frequency point (as presented earlier in Fig. 11), the total time taken to perform phase 1 of the method is few hours. Note that phase 1 is performed only once for a system at design time and the timing overhead for this is negligible.

Furthermore, we can propose some efficient algorithms to solve the proposed optimization problems. In Appendix A, we present Algorithm 1 as a formal description of a polynomial time solution for the proposed optimization problem in Eqs. (53)–(62).

7. Proposed application-aware mapping technique

Based on the prepared hardware (performance counters and PCU) required for the proposed runtime reconfiguration approaches, in Section 6, we propose a mapping technique for the target 3D CMP to improve the thermal distribution at runtime as an essential need in future many-core CMPs.

Empirically, we have observed that if a memory bounded- application/thread has a smaller amount of cache than needed, it will result in more cache misses and lower IPC. Therefore, IPC can be a suitable parameter to identify between memory and computation-bounded applications/threads [74].

Since computation-bounded applications/threads create hot-spots on the chip [75], in the proposed runtime mapping technique, we place the memory and computation-bounded applications/threads on the core layer, uniformly, to balance the temperature. The proposed mapping technique predicts the CPI (or IPC) of different applications/threads for the next time interval by collecting performance counters data at runtime and places tasks with complementary characteristics on adjacent neighbors to balance the temperature. The proposed mapping technique includes two stages, inter-region and intra-region mapping. In this technique, there is a CPI predictor component in each core tile which has an important role. The overview of the proposed technique is shown in Fig. 18. The PCU is responsible for doing this mapping technique at the end of each predefined time interval.

7.1. Proposed CPI predictor

We equip each core-tile in the core layer with a CPI predictor. The CPI-predictors use prepared performance counters in each core-tile. The goal of the CPI predictor, shown in Fig. 18, is to determine $CPI_{T+1}(i, j)$, for all $j \in [1, n]$, for the next time interval. Let $CPI_{T+1}(i, j)$ be the CPI of application/thread i on core j in the next time interval in Eq. (67). To predict $CPI_{T+1}(i, j)$, the CPI predictor component uses CPI information measured by the hardware counter broken down into two components: compute CPI (base CPI in the absence of miss events), and memory CPI (cycles lost due to misses in the cache hierarchy). With these measurements on core j, we predict the CPI for the next time interval using a linear predictor as follows:

$$CPI_{T+1}(i, j) = \delta_j^{com} \cdot CPI_T^{comp}(i, j) + \delta_j^{mem} \cdot CPI_T^{mem}(i, j) \tag{67}$$

where δ_j^{com} and δ_j^{mem} are fixed parameters that are computed for each core configuration at off-line. At the end of each time interval,

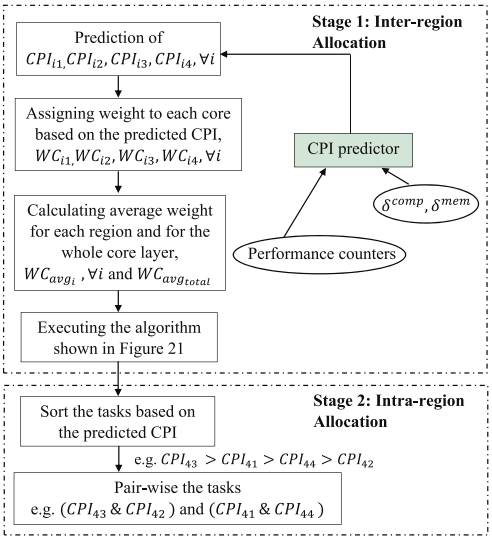


Fig. 18. Overview of dynamically adaptive mapping approach for a 3D CMP.

$$Predicted\ CPI = \begin{cases} CPI > t_1; & HM \\ t_1 < CPI < t_2; & MM \\ t_2 < CPI < t_3; & M \\ t_3 < CPI < t_4; & MC \\ CPI < t_5; & HC \end{cases}$$

Fig. 19. Allocation of weight to each application/thread.

each core in the core layer sends predicted CPI by the CPI predictor in a control flit over VIPs to the monitoring tile, as shown in Fig. 13. After gathering the CPI control flits, the PCU starts the mapping algorithm.

According to the predicted CPI of each application/thread for the next epoch, if the predicted CPI is greater than a threshold introduced and set up in [74], that application/thread is in the range of memory-intensive loads. Also, if the predicted CPI is less than the threshold, that application/thread is in the range of computation-intensive loads. Therefore, based on the predicted CPI for each core, the PCU decides to allocate applications/threads on to cores in two stages.

7.2. Stage 1: inter-region mapping

As shown in Fig. 19, we set up five thresholds, {t_1, t_2, t_3, t_4, t_5}, and based on the predicted CPI, the applications/threads are classified into five types and a weight to each core is assigned: Heavy memory-intensive (HM), Medium Heavy memory-intensive (MM), Medium (M), Heavy computation-intensive (HC), and Medium Heavy computation-intensive (MC). Then, the PCU calculates the average weight for each region (WC_{avg_i}) and for the whole core layer ($WC_{avg_{total}}$) as shown in Fig. 20.

Based on an algorithm shown in Fig. 21, the PCU compares each WC_{avg_i} with $WC_{avg_{total}}$ to see the difference. If the difference between WC_{avg_i} and $WC_{avg_{total}}$ is more than a threshold, the highest-weight application/thread in the region with the largest WC_{avg_i} and the lowest-weight application/thread in the region with the smallest WC_{avg_i} are swapped to balance thermal distribution. This process is iterated until the difference between each WC_{avg_i} and $WC_{avg_{total}}$ is under the threshold. The PCU conducts application/thread migration if needed after the iteration converges.

7.3. Stage 2: intra-region mapping

In the proposed mapping technique, we assume that each region on the core layer includes four cores. In this stage, the PCU

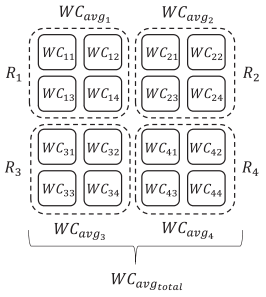


Fig. 20. Computing of $\boldsymbol{WC_{avg_i}}$, $\{\boldsymbol{i = 1, 2, 3, 4}\}$, and $\boldsymbol{WC_{avg_{total}}}$ for a core layer with 16 cores.

While ($|WC_{avg_i} - WC_{avg_{total}}| \geq threshold, \forall i$) {

1. Sort (WC_{avg_i}) ;
2. Find the application/thread with the largest weight in the region which has the largest WC_{avg_i} and name it T1;
3. Find the application/thread with the smallest weight in the region which has the smallest WC_{avg_i} and name it T2;
4. Swap T1 and T2;
5. Compute the new $WC_{avg_i}, \forall i$

}

Fig. 21. Scalable inter-region mapping algorithm.

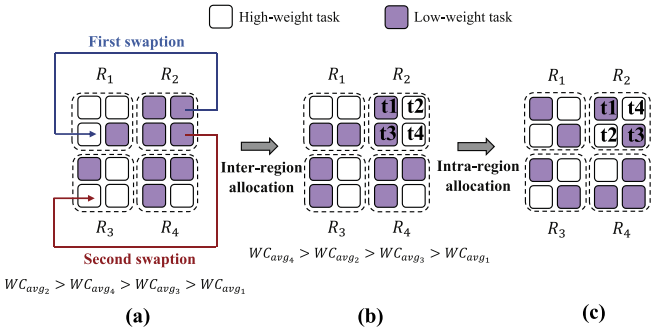


Fig. 22. An example of the two-stages application mapping in a 16-core layer of a 3D CMP.

sorts the applications/threads with respect to their predicted CPI and groups them in pairs by selecting the highest and lowest ones from the remaining sorted as ($CPI1 > CPI2 > CPI3 > CPI4$). For example in this inequality, the algorithm pairs (task1, task4) and (task2 and task3) on adjacent neighbors, in this stage.

Since computation-bounded applications/threads create hot-spots on the chip [75], based on using this two-stages mapping technique, we try to place memory and computation-bounded applications/threads on adjacent neighbors to balance the temperature on the chip.

Fig. 22 shows a simple example of the two-stages application mapping in a core layer of a CMP system with 16 cores. In this example, there are 4 regions; the regions are named R1, R2, R3, and R4. Regions R2 and R4 initially have 4 and 3 high-weight tasks, respectively, while regions R1 and R3 only have 1 high-weight task each. In this example, $WC_{avg_2} > WC_{avg_4} > WC_{avg_3} > WC_{avg_1}$. According to the algorithm in Fig. 21, the task with the highest weight in R2 is swapped with the task with the lowest weight in R1, and we get $W_{avg_2} > W_{avg_4} > W_{avg_1} > W_{avg_3}$. Similarly, the algorithm swaps the new highest-weight task in R2 with the lowest-weight task in R3 and gets the difference between each WC_{avg_i} and $WC_{avg_{total}}$ under the threshold. After the inter-region reallocation, the algorithm performs the intra-region application mapping to finalize the location of each task. In this stage, for example in the region R2, there is $CPI_{21} > CPI_{23} > CPI_{22} > CPI_{24}$, then the algorithm pairs task1 and task4, also, task2 and task3 on adjacent neighbors as shown in Fig. 22(c) to balance the temperature.

As shown in Fig. 22(c), all of high-weight and low-weight tasks have been distributed uniformly on the core layer and in contrast to Fig. 22(a), the center of the core layer will not be able to create a hot-spot region.

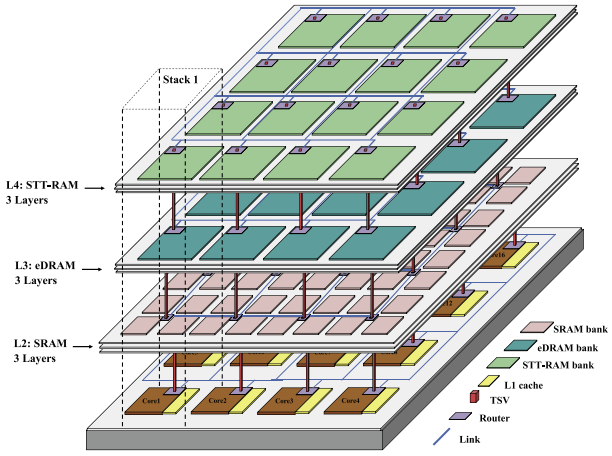


Fig. 23. A 16-core CMP architecture with three level heterogeneous cache hierarchy used for the evaluation.

7.4. Applying two-stages mapping technique at runtime

In this sub-section, we can integrate each of the proposed optimization-based reconfiguration schemes in Section 6 and proposed runtime mapping technique. As a first step, we assign n applications/threads to n cores in the core layer in a random manner and assign a limited number of cache banks in each level of the cache hierarchy to each core, then start running the applications/threads for a fixed time (e.g., 20 ms). Based on collected data in performance counters and providing feedback to the predictor component, the CPI parameter for each core is predicted and the proposed two-stages mapping technique, shown in Fig. 18, is started by the PCU. After placing applications/threads on the cores decided by the two-stages mapping in the PCU, the proposed reconfiguration scheme, in Section 6, is applied every *reconfiguration time interval*, as shown in Fig. 11.

In order to improve the temperature distribution of the 3D CMP in presence of workload changes, the PCU applies the proposed runtime mapping technique every $20 \times$ *reconfiguration time interval*.

8. Experimental evaluation

8.1. Experimental platform

In order to validate the efficacy of 3D CMP architectures in this work, we employ a detailed simulation framework driven by traces extracted from real application workloads running on a full-system simulator. The traces have been extracted from the GEM5 full-system simulator [62]. For simulating a 3D CMP architecture, the extracted traces from GEM5 are interfaced with 3D Noxim, as a 3D NoC simulator [61].

A full-system simulation of a 16-core CMP architecture at 32 nm technology, as shown in Fig. 23, is performed for evaluation in this work. Fig. 23 shows more detail in compared to the Fig. 1 for a 16-core CMP. As shown in this figure, in the core layer, each core is connected to a router. The area of the core tile (consisting of a processing core, private 32KB L1 instruction and data caches, and

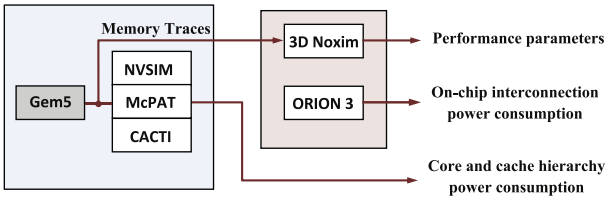


Fig. 24. Platform setup used in this work.

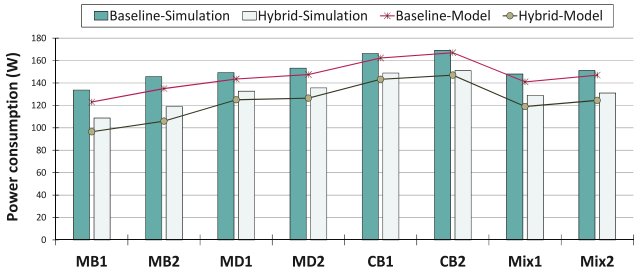


Fig. 25. Validation of the power consumption model under executing multiprogramed workloads.

the cache controller with cache tags) is 3.5 mm^2 estimated by McPAT [60] and CACTI 6.0 [59]. The detailed system configurations are given in Table 4.

GEM5 is augmented with McPAT and 3D Noxim with ORION [64] to calculate the power consumption in this paper. The cache capacities and energy consumption of SRAM and NVMs are estimated from CACTI and NVSIM [58], respectively. The simulation platform for the evaluation of our proposed and other 3D CMP architectures in this work is illustrated in Fig. 24.

For experimental evaluation, maximum temperature limit and dark-silicon peak power budget, T_{max} and P_{budget}, is assumed to be 80 °C and 100 *W*, respectively.

We assumed that the area of each cache layer is equal to the area of the core layer. As shown in Fig. 23, in a core-stack, at the first level of the cache hierarchy, there are four SRAM cache banks, each of which has 256KB capacity, placed directly on top of one core. Based on estimates from CACTI 6.0, a 1MB SRAM cache bank and associated router/controller have an area roughly equal to the area of one core. At the second and third levels of the cache hierarchy, there is a 4MB eDRAM and a 4MB STT-RAM cache bank placed directly on top of one core in each layer, respectively. We consider three layers in each cache level of the stacked hierarchy for the 16-core CMP, as shown in Fig. 23. Estimations from CACTI and NVSIM shows that eDRAM and STT-RAM are about four times denser than that of SRAM in the same area. The detailed properties for heterogeneous cache memories in different technologies are listed in Table 1.

For 3D temperature estimation, we employed HotSpot [63] version 5.0 as a grid-based thermal modeling tool. The thermal resistance and capacitance of inter-layer material, (i.e. R_{inter} and C in Fig. 17) are 140.4 J/K and 0.1 K/W, respectively. The interface material between two adjacent silicon layers are connected by homogeneous TSVs.

We use multiprogramed workloads consisting of 16 applications for performing our experiments. The applications are selected from the SPEC2000/2006 benchmark suites [49]. Based on memory demand intensity of benchmark applications, we classified them into three groups: memory bounded, medium, and computation bounded benchmarks. From this classification, we generated a range of workloads (combinations of 16 benchmarks), as summarized in Table 5. Note that, the number in parentheses is the number of instances. Moreover, we use multithreaded workloads selected from PARSEC [68] for performing the multithreaded experiments.

In our evaluation, in Section 8.2, first, we experiment the proposed power model and runtime optimization-based reconfiguration techniques using a random mapping, Figs. 25–39. Under a random mapping approach, applications/threads are randomly mapped to the cores. In Section 8.2.2, we have some sensitivity analysis, Figs. 35–36 and Figs. 38–39. Finally, in Section 8.2.3, we integrate the proposed runtime mapping technique and optimization-based reconfiguration approaches and study the effect of applying the mapping algorithm in parallel with the reconfiguration techniques on the temperature, Figs. 40 and 41. Base on this trend, we study about the proposed techniques in Sections 6 and 7, step by step.

8.2. Experimental results

8.2.1. Experimental results for the proposed runtime optimization-based reconfiguration techniques

In this sub-section, we evaluate a 3D CMP with stacked cache hierarchy as shown in Fig. 23 in three different cases: a CMP with fixed core frequencies in the core layer and SRAM-only cache levels in the stacked hierarchy with fixed capacity (Baseline), the CMP with fixed core frequencies and fixed maximum available capacity at each level of the heterogeneous cache hierarchy (Hybrid-fix), and the proposed CMP with runtime heterogeneous cache hierarchy management and per core DVFS (Hybrid-Proposed). As described in Section 6, since there are limited power budget and temperature in modern processors, for better performance in the Baseline and Hybrid-fix we use allocated cache banks in each level in the shared manner. In the Hybrid-Proposed scheme, we partition the shared cache space according to the specific demanding of each individual application in a workload set.

Based on Fig. 23, in Hybrid-Proposed the capacity of L2 cache is $16 \times 4 \times 256$KB SRAM, the capacity of L3 cache is 16×4MB eDRAM and the capacity of L4 cache is determined 16×4MB STT-RAM, respectively. Hybrid-Fix is similar to the Hybrid-Proposed in cache levels capacity and number of layers in each level. Since STT-RAM and eDRAM is made four times denser than SRAM, the Baseline architecture with $16 \times 4 \times 256$KB SRAM in L2 level, 16×1MB SRAM in L3 level and 16×1MB SRAM in L4 level takes up similar amount of area to Hybrid-Fix and Hybrid-Proposed.

According to the filling of the look up table (shown in Fig. 12) used in the Hybrid-Proposed scheme based on solving the optimization problem in Eqs. (43)–(52) or optimization problem in Eqs. (53)–(62), we categorized it into two different schemes: Hybrid-Proposed1 and Hybrid-Proposed2, respectively. In the Hybrid-Proposed1, optimization variables are resultant of solving the optimization problem in Eqs. (43)–(52) and in the Hybrid-Proposed2, optimization variables are resultant of solving the optimization problem in Eqs. (53)–(62).

According to our results acquired by McPAT and Synopsys Design Compiler, the designed controller has merely 74.88 ns latency and consumes less than 1.63 mW for the reconfiguration operation in each time interval. Furthermore, our controller has a negligible area overhead, even though in the dark silicon era, area overhead is of less importance.

In addition to the comparing of Hybrid-Proposed1 and Hybrid-Proposed2 with Baseline and Hybrid-fix, we compare them with the newest proposed 3D stacked cache architecture, 3D-CRP [65]. Recently, Meng et al. [65] proposed a 3D cache resource pooling (3D-CRP) architecture and a runtime management policy. They designed an integrated cache management and job allocation policy that maximizes the energy efficiency of 3D systems for dynami-

Table 4
Specification of CMP configurations evaluated in this work.

Component	Description
Number of cores	16, 4×4 mesh
Core configuration	Alpha21164, 3 GHz, area 3.5 mm^2, 32 nm
L1 cache	SRAM, 4 way, 32B line, size 32KB per core
L2/L3/L4 caches	L2: SRAM, L3: SRAM, L4: SRAM (Baseline) L2: SRAM, L3: eDRAM, L4: STTRAM (Hybrid)
Main memory	4GB, 320 cycle access, 4 on-chip Memory Controllers at each corner node
Network router	2-stage wormhole switched, XYZ routing, virtual channel flow control, 2 VCs per port, a buffer with depth of 5 flits per each VC, 8 flits per Data Packet, 1 flit per address packet, each flit is set to be 16-byte long
Network topology	3D network, each layer is an 4×4 mesh, each node in layer 1 has a router, 16 TSV links which are128b bi-directional in each layer

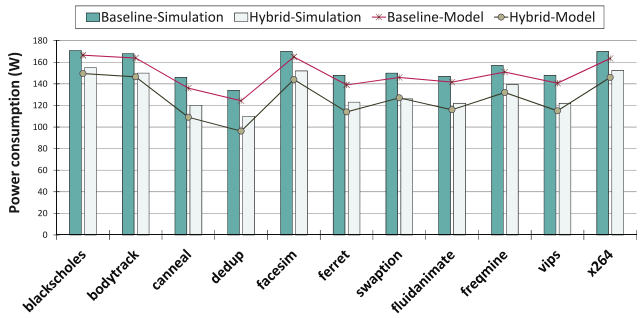


Fig. 26. Validation of the power consumption model under executing multithreaded workloads.

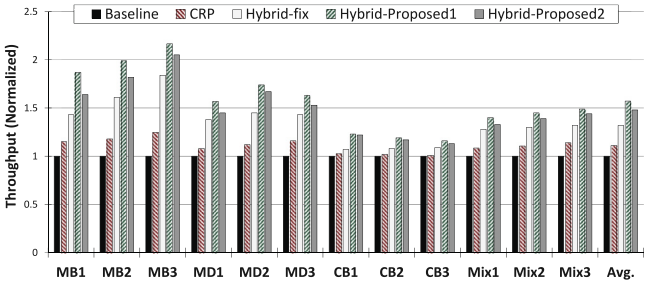


Fig. 27. Comparison of throughput results of multiprogramed workloads normalized to the Baseline.

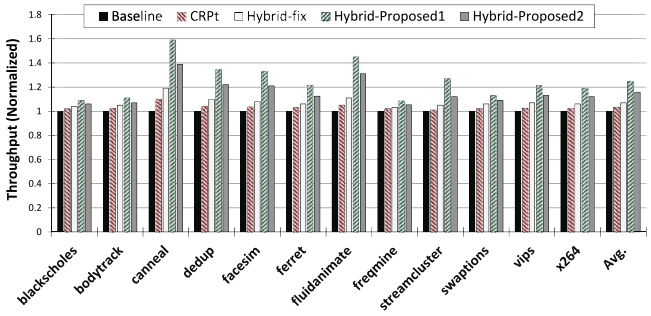


Fig. 28. Comparison of throughput results of multithreaded workloads normalized to the Baseline.

cally changing workloads [65]. Their policy predicts the resource requirements by collecting the performance characteristics of each workload at runtime. The 3D stacked cache systems in their work is based on homogenous traditional memory technologies and they don't consider maximum power budget and temperature limit as the most important constraints in the proposed technique. Also, their proposed architecture only considers multiprogramed workloads. Specification of the CRP architecture used in our work is same as Table 4. The characteristics of 3D stacked cache system in the CRP has been assumed like Baseline architecture.

First, in Figs. 25 and 26, we validate the proposed power model in Section V for a homogenous (Baseline) and heterogeneous architecture (Hybrid-fix). These figures show that the proposed power model estimates the power consumption of heterogeneous and homogenous 3D CMPs, with a good degree of accuracy, under running both multiprogramed and multithreaded workloads. As we see, in the computation intensive workloads, our model and simulation are near, while in the memory intensive workloads, they have some differences.

Figs. 27 and 28 show the results of normalized throughput for multiprogramed and multithreaded workloads, respectively, where throughput is the number of executed instructions per second (IPS). As shown in Fig. 27, Hybrid-Proposed1 yields up to 0.39% throughput improvement compared with the Hybrid-fix. Similarly, Hybrid-Proposed2 yields up to 25.2% throughput improvement compared with the Hybrid-fix. It is because of the lower static power consumption and larger cell density of NVM technologies that Hybrid-Proposed1 and Hybrid-Proposed2 allocates larger cache capacity in each level of the hierarchy and suitable clock frequency to the cores for memory-intensive workloads under temperature constraint. In this regard, Hybrid-Proposed1 yields up 16% throughput improvement, compared with the Hybrid-Proposed2. It is because of that Hybrid-Proposed2 maximizes performance under the fixed power budget constraint. Allocating suitable frequency/voltage to cores under power and temperature constraints in Hybrid-Proposed methods and, also, according to the lower static power consumption and larger cell density of NVM technologies, they work better than CRP technique. Hybrid-Proposed1 and Hybrid-Proposed2 improve throughput by about 40% and 32.3% on average, respectively, in comparison with the CRP architecture. Moreover, Hybrid-Proposed1 and Hybrid-Proposed2 improve throughput by about 54.3% and 48.2% on average, respectively, in comparison with the Baseline architecture.

Since CRP technique is applied for multiprogramed workloads [65], we changed it and created a new version named CRPt for multithreaded workloads. As shown in Fig. 28, Hybrid-Proposed1 and Hybrid-Proposed2 improve throughput by about 20% and 12%, on average, compared with the CRPt technique. Further, Hybrid-Proposed1 and Hybrid-Proposed2 improve throughput by about 25% and 16%, on average, compared with the Baseline architecture.

Figs. 29 and 30 show the results of normalized energy efficiency for multiprogramed and multithreaded workloads, respectively, where energy efficiency is energy-delay product (EDP). Since the Hybrid-Proposed1 and Hybrid-Proposed2 regulate the frequency/voltage of the cores and activate larger number of cache levels due to the lower static power and larger cell density of NVM technologies, finish its execution earlier than Hybrid-fix and Baseline schemes. As shown in Fig. 29, Hybrid-Proposed1 and Hybrid-Proposed2 yield about 26.8% and 20.2% EDP reduction on average,

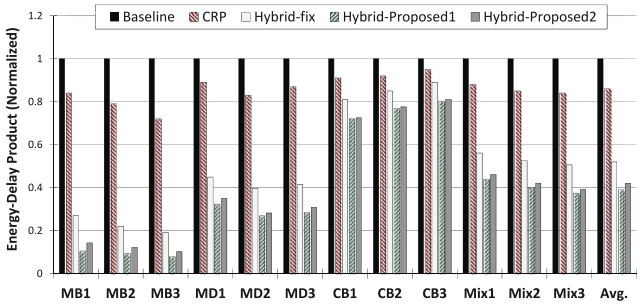


Fig. 29. Comparison of Energy-Delay Product (EDP) results under multiprogramed workloads execution normalized to the Baseline.

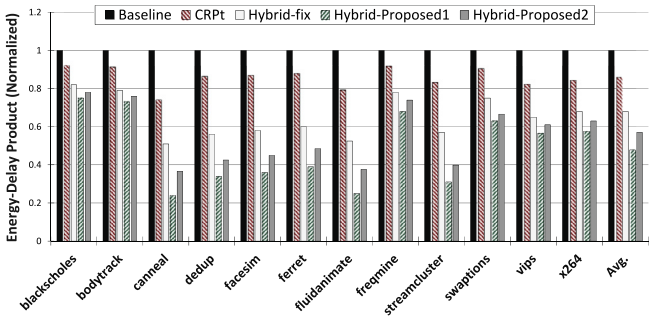


Fig. 30. Comparison of Energy-Delay Product (EDP) results under multithreaded workloads execution normalized to the Baseline.

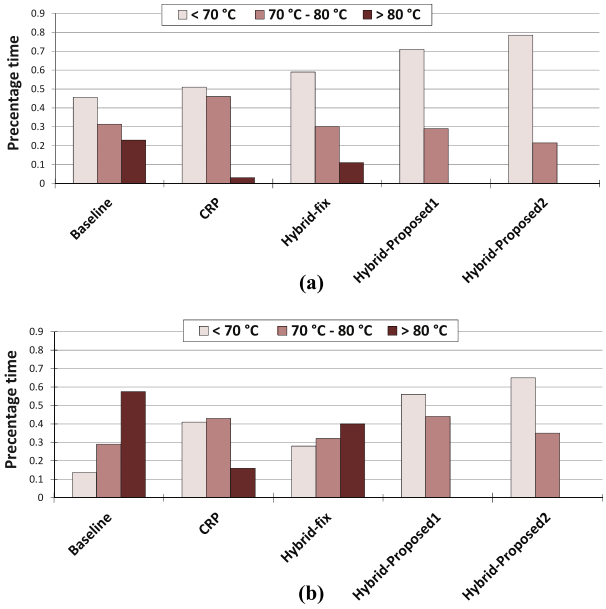


Fig. 31. Comparison of percentage time spent on average by the up layer of each core-stack at different temperature points, (a) under executing MB2 test program suite, (b) under executing CB2 test program suite.

respectively, in comparison with the Hybrid-fix. Moreover, Hybrid-Proposed1 and Hybrid-Proposed2 reduces EDP by about 52% and 48.5% on average, respectively, compared with the CRP. In conclusion, Hybrid-Proposed1 and Hybrid-Proposed2 improves 61% and 58% EDP reduction on average, respectively, in comparison with the Baseline architecture.

Fig. 30 shows that Hybrid-Proposed1 and Hybrid-Proposed2 reduces EDP by about 46% and 35% on average, respectively, compared with the CRPt. Moreover, Hybrid-Proposed1 and Hybrid-Proposed2 improves 52.2% and 43% EDP reduction on average, respectively, compared with the Baseline architecture.

Fig. 31(a) shows the percentage time that the top layer of the 3D CMP system in each case, on average, spent at different temperature points under executing MB2 test program suite. As shown in the figure, the Hybrid-Proposed1 and Hybrid-Proposed2 methods always ensures that the up layer of the proposed 3D CMP are below the maximum temperature of 80 °C, while the Baseline, CRP and Hybrid-fix spend a significant amount of time above the maximum temperature. As shown in Fig. 31(b), for CB2 test program suite as one of the computation intensive suite, the Baseline, CRP and Hybrid-fix schemes spend up to 57.6%, 16% and 40% of the time above the maximum temperature, respectively.

Fig. 32(a) and (b) show the comparison of maximum temperature of the up layer of each core-stack of the 3D CMP system in each case under executing MB2 and CB2 test program suites, respectively. As shown in the figures, maximum temperature of the up layer of each core-stack in the Hybrid-Proposed1 and Hybrid-Proposed2 methods always remain below the maximum temperature of 80 °C, while the Baseline, CRP and Hybrid-fix schemes violates the maximum temperature constraint.

Fig. 33 plots the runtime profile of power consumption of the 3D CMP system in each case when executing MB2 test program suite under power budget constraint.

As shown in Fig. 33, the purple line illustrates the maximum power budget for the system (i.e. TDP). As can be observed in this figure, the power consumption in some cases exceeds the TDP in Baseline and Hybrid-fix schemes. As shown in this figure, Hybrid-proposed2 never allows the total power consumption to violate the TDP.

Due to the limitation of space, we only present the Figs. 31–33 for MB2 and CB2 test program suites. All the test program suites used in this paper, multiprogramed and multithreaded, were experimented and in all of them, Hybrid-Proposed1 and Hybrid-Proposed2 did not violate the temperature and power budget constraint.

In Fig. 34, the lifetime results of each test program suite for Baseline and Hybrid-Proposed2 schemes has been calculated. This calculation has been done without consideration to the aging mechanisms such as Negative Bias Temperature Instability (NBTI) and other temperature-related problems in reliability which is beyond the scope of current work. Due to the higher leakage power consumption of SRAM cache which leads to increase temperature, SRAM cache in the 3D architecture suffers from NBTI degradation significantly more than the non-volatile memories. In a 3D architecture, higher temperature in the core layer makes cache layers hotter and thus imposes more degradation in the upper levels. The temperature variation aggravates the NBTI effects in the upper layer.

To evaluate lifetime, we assumed that programs in a test program suite continuously run until one of the cache blocks exceeds the maximum number of endurable writes in each cache level. We assumed the endurable maximum number of writes for SRAM, DRAM, STT-RAM and PRAM based on Table 6 [56]. As shown in Fig. 34, the life time of the Baseline architecture is up to 7.5X and 1.85X on average in comparison to the Hybrid-Proposed2, because of the low endurance problem of NVM technologies.

8.2.2. Sensitivity analysis

In this sub-section, we evaluate the scalability of our proposed schemes using the Mix2 multiprogramed workload, as one of the moderate workloads in Table 5, under a random mapping. Note that in all of the sensitivity studies, the architecture of the stacked cache hierarchy (number of levels, layers and technology of each cache level) is same as Fig. 23.

a) Sensitivity to the different number of cores:

Table 5
Multiprogramed workloads used in the experiment.

Test program suite	Benchmarks
Memory Bounded set1 (MB1)	zeusmp(2), libquantum(2), lbm(2), GemsFDTD(2), art(4), swim(4)
Memory Bounded set2 (MB2)	zeusmp(3), libquantum(3), lbm(3), GemsFDTD(3), art(2), swim(2)
Memory Bounded set3 (MB3)	zeusmp(4), libquantum(4), lbm(4), GemsFDTD(4)
Medium set1 (MD1)	mcf(2), sphinx3(2), bzip2(2), calculix(2), leslie3d(2), gcc(2), cactusADM, milc, omnetpp, wupwise
Medium set2 (MD2)	mcf(3), sphinx3(3), leslie3d(2), gcc(2), cactusADM(2), milc(2), omnetpp(2)
Medium set3 (MD3)	mcf(2), sphinx3(2), bzip2, calculix, leslie3d(2), gcc(2), cactusADM(2), milc(2), omnetpp(2)
Computation Bounded set1 (CB1)	parser(2), applu(2), face_rec(2), equake(2), astar(2), hmmer(2), bzip2(2), calculix(2)
Computation Bounded set2 (CB2)	parser(2), applu(2), face_rec(2), equake(2), astar(2), hmmer(2), bzip2, calculix, mpeg_dec(2)
Computation Bounded set3 (CB3)	parser(2), applu(2), face_rec(2), equake(2), astar(2), hmmer(2), mpeg_dec(4)
Mixed set1 (Mix1)	sphinx3, mcf, astar(2), hmmer(2), gamess(2),perlbench(2), gromacs(2), tonto(2), gcc, leslie3d
Mixed set2 (Mix2)	sphinx3(2), mcf, astar(2), hmmer, gamess(2), perlbench(2), soplex, gromacs, gcc(2), leslie3d(2)
Mixed set3 (Mix3)	sphinx3(2), mcf(2), astar, hmmer, gamess, perlbench, soplex(2), gcc(3), leslie3d(3)

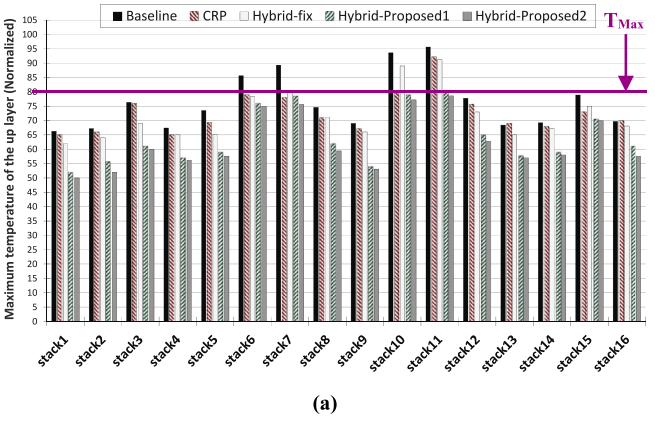


Fig. 32. Comparison of maximum temperature of the up layer of each core-stack under temperature limit constraint, (a) under executing MB2 test program suite, (b) under executing CB2 test program suite.

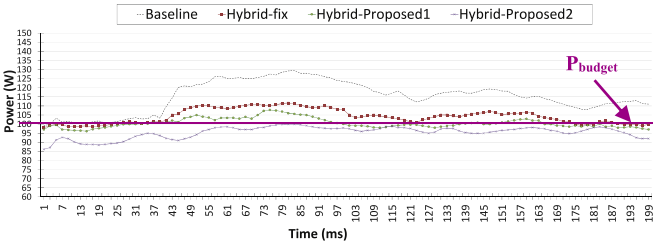


Fig. 33. A runtime profile of the power consumption, executing MB2 test program suite, under power budget constraint.

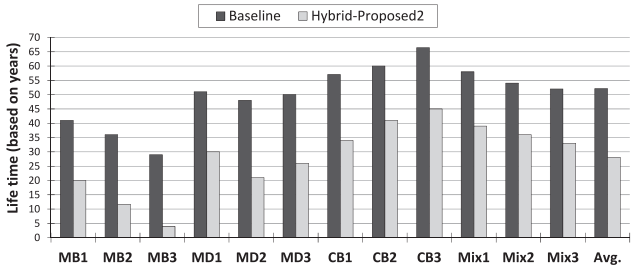


Fig. 34. Comparison of the life time results of each test program suite for Baseline and Hybrid-Proposed2 schemes.

Table 6
The endurable maximum number of writes for various memory technologies.

Technology	SRAM	eDRAM	STT-RAM	PRAM
Endurance	10^{16}	10^{16}	4×10^{12}	10^{9}

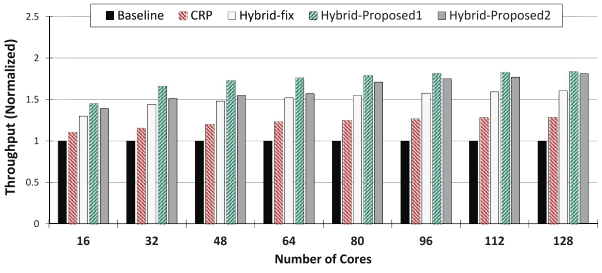


Fig. 35. Comparison of throughput results normalized to the Baseline.

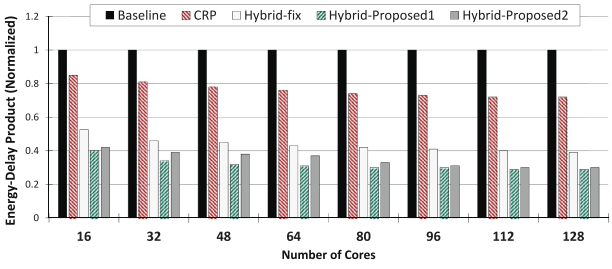
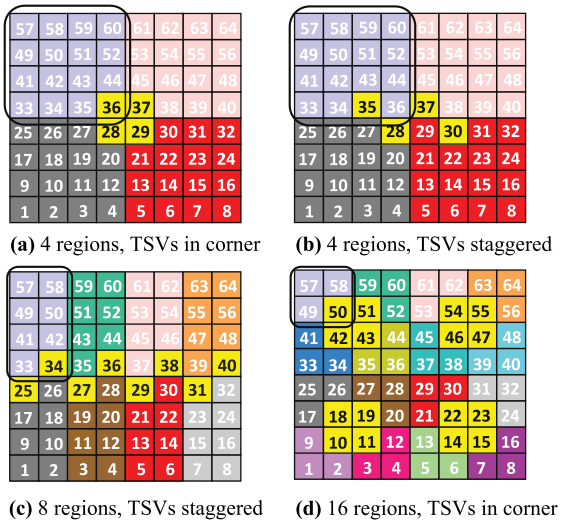


Fig. 36. Comparison of Energy-Delay Product (EDP) results normalized to the Baseline.

In this section, we analyze the sensitivity of our proposals to the different number of cores. Figs. 35 and 36 show the results of normalized throughput and energy efficiency, respectively, for 16, 32, 48, 64, 80, 96, 112, 128-core CMPs. As shown in Fig. 35, Hybrid-Proposed1 and Hybrid-Proposed2 have better throughput than the



(a) 4 regions, TSVs in corner **(b)** 4 regions, TSVs staggered

(c) 8 regions, TSVs staggered **(d)** 16 regions, TSVs in corner

Fig. 37. Illustration of the various placement of TSVs and the sub-division of core layer into regions.

Baseline, CRP and Hybrid-fix for each platform. Furthermore, Figs. 35 and 36 show that the proposed methods' throughput and EDP improvements over the Baseline increase with increasing the number of cores.

In our simulation, for clear explanation about the mapped workloads, the number of cores in each platform are coefficient of 16. For example in a 64-core system, we ran four Mix2 sets in a random distribution.

b) Sensitivity to the placement and the number of TSVs:

In this section, we analyze the sensitivity of our proposals to the number and placement of TSVs. For this sensitivity study, we first start by experimenting with the placement of the TSVs in a 64-core CMP. Fig. 37 shows the three different placements that we experimented with: corner, staggering and All-TSVs. In the corner placement, TSVs are placed at the corner of the regions, while in the staggering placement, TSVs are placed staggered. Additionally, we sub-divided the core layer into 4, 8 and 16 regions by different number of TSVs in the corner and staggering placements. In the corner and staggering placements, based on XYZ routing used in the proposed architecture, packets are serialized to routers in each region. Hence, a packet routed from the core layer to the cache layer is first routed using XY routing to a particular router in the core layer, followed by a TSV traversal in the vertical direction to a router in the related cache region. Finally, the packet follows XY routing again in the cache layer to the destination cache bank. This scheme repeats when communicating between the cache layers to the core layer. In the All-TSVs option, we designate one TSV in each core tile in the core layer (64 TSVs) through which all cores can communicate their request packets with any cache bank in the corresponding region in the cache layer.

Fig. 38 shows the result of this study. This figure shows that in the Hybrid-Proposed1 there is 5% improvement, on average, in throughput with the staggering placement of TSVs compared to the corner. It is because of that in the staggering placement, when using XY routing in the core layer, the Y-direction flows going to the TSVs do not overlap with each other because the TSVs are now placed along different columns. Furthermore, Fig. 38 shows that with increasing number of TSVs because of reducing congestion, throughput improvement increases. In this figure, throughput results have been normalized to All-TSVs. Note that experimental evaluation in the Section 8.2.1 have been done based on All-TSVs, as shown in Fig. 23.

c) Sensitivity to the different number of DVFS levels:

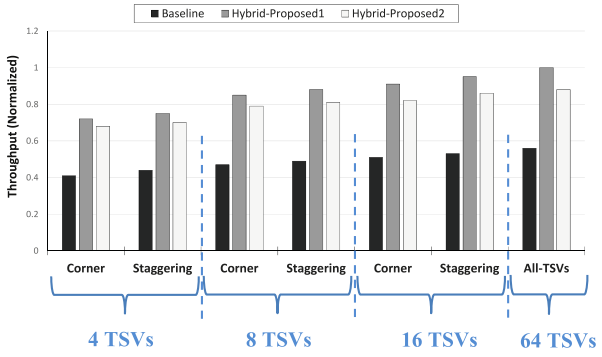


Fig. 38. Sensitivity to placement and number of TSVs.

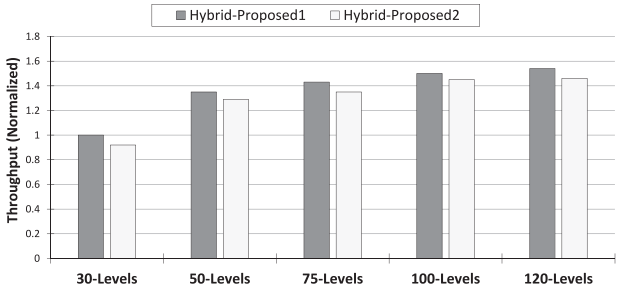


Fig. 39. Sensitivity to different number of DVFS levels.

In this section, we study about the effect of the different number of DVFS levels, used for filling the look up table shown in Fig. 12, on the performance of our proposals. For reducing the distance between estimated frequency/voltage by the proposed optimization problems and supported values in the look up table, we can increase the number of DVFS levels and start our sensitivity study. As shown in Fig. 39, experiments have been done for five different number of levels. Fig. 39 shows that throughput improvement increases from 30-Levels to 75-Levels. As shown in this figure, the percent of improvement from 75-Levels to 120-Levels is not impressive and leads to using large lookup tables. In this figure, throughput results have been normalized to 120-Levels. Note that experimental evaluation in the Section 8.2.1 have been done based on 50-Levels.

8.2.3. Experimental results for the integration of the proposed mapping technique and optimization-based reconfiguration schemes

In this subsection, we integrate the proposed runtime mapping technique in Section 6 and the optimization-based reconfiguration approaches in Section 7, and study the effect of applying the mapping algorithm in parallel with the reconfiguration techniques on the temperature. For this study, we experiment the results of this integration on the temperature in a 64-core CMP with 9 layers heterogeneous cache hierarchy based on the Mix2 multiprogramed workload such as Section 8.2.2. Figs. 40 and 41 illustrate the spatial distribution of the temperature of the 5th layer (middle layer) of the cache hierarchy under the random and proposed mapping technique, respectively. As shown in Fig. 40, the distribution of the temperature is non-uniform under random mapping. Fig. 41 shows that temperature distribution is more uniform compared with Fig. 40 and it is because of that the proposed mapping technique places hotspot computation intensive applications in the neighborhood of memory intensive applications and alleviates the surrounding hotspots placed based on the random mapping. If two or more hotspots come close, this will produce thermal coupling and therefore locally raise the temperature. In our proposed map-

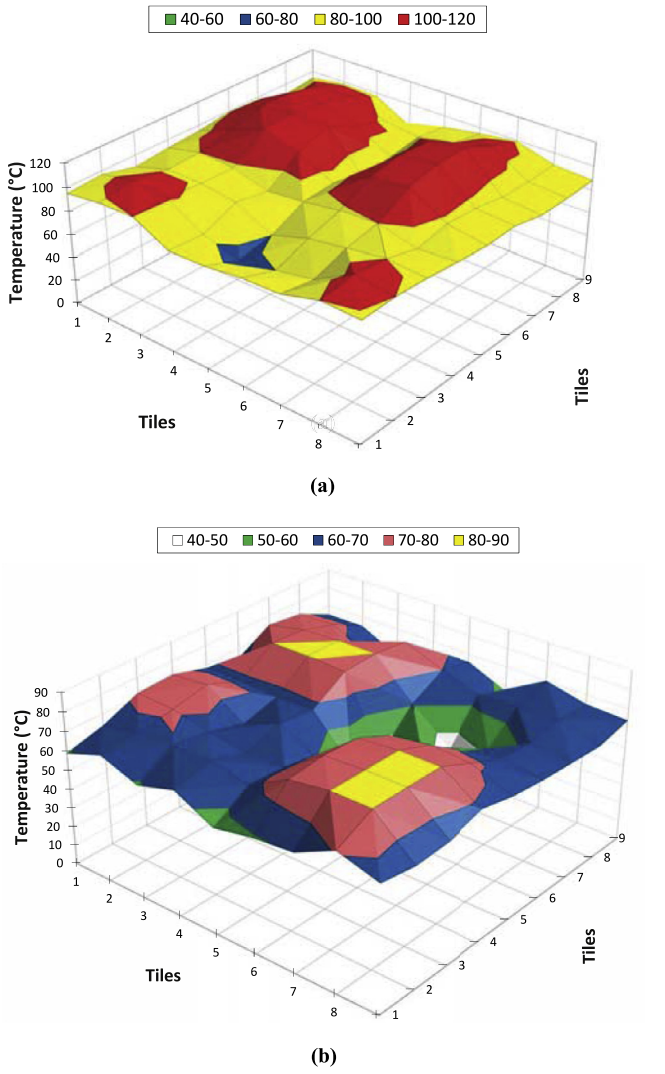


Fig. 40. Thermal maps of the middle layer of the cache hierarchy in a 64-core CMP with random mapping, (a) Baseline, (b) Hybrid-Proposed1.

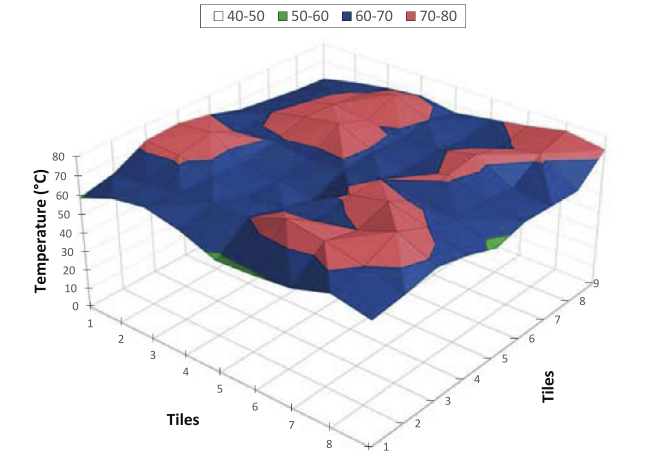


Fig 41. Thermal map of the middle layer of the cache hierarchy in a 64-core CMP under integration of the Hybrid-Proposed1 and the proposed mapping technique.

ping, interleaving hot and cold blocks in an effective method helps to provide lower global power density (therefore, lower temperature). In Fig. 41, we have 3 °C colder hotspots compared with Fig. 40(b).

9. Conclusion

In this paper, we proposed an integrated solution of runtime cache way assignment (i.e., cache capacity allocation and power gating of unnecessary cache ways) and per-core DVFS for CMPs with 3D stacked hybrid cache hierarchy in order to maximize the system performance in an energy-efficient and temperature-aware manner for both of multiprogramed and multithreaded workloads. We proposed analytical formulations and applied them as a runtime solution for future 3D CMPs. In continue, we proposed an application aware mapping technique which was integrated with the proposed runtime optimization-based reconfiguration mechanisms efficiently to balance the temperature distribution. Experimental results show that the proposed method (i.e., Hybrid-Proposed1) improves the instruction throughput and energy-delay product by about 54.3% and 61% on average, respectively, in comparison with the conventional method where homogenous cache technology is used.

Appendix A

Algorithm 1 is a formal description of a polynomial time solution for the proposed optimization problem in Eqs. (53)–(62). In this algorithm, $L_{i,k}^{max}$ is maximum number of layers at the kth cache level on the core layer.

The order of this algorithm is $O(m.n)$ in a CMP with n cores and m steps in each time interval. Searching can be exhaustively done (in polynomial time) to determine the best number of cores,

Algorithm 1 Optimal frequency assignment, cache hierarchy reconfiguration and core selection.

1. $J1^* \leftarrow \infty$, $L_{i,k}^{max} \leftarrow L_k^{max}$
2. **for** $i \in [1, n]$ **do: for** $k \in [1, N]$ **do:** $L_{i,k} \leftarrow 0$
3. **for** $t \in [1, 100]$ **and** m=200 **do:**
4. **for** $i \in [1, n]$ **do:**
5. **if** $(t = 1)$ **then:** $(T_i^{core})_{init} \leftarrow T_{start}$
6. **else:** $(T_i^{core})_t \leftarrow T_{start} + R_{hs} \cdot (P_i^{core})_t$
7. $(f_i)_t^* \leftarrow f_{max}$ **and** $\theta \leftarrow 1$
8. $\Delta \leftarrow P_{budget_i} - (P_{max} .((f_i)_t^{\alpha}/f_{max}^{\alpha}) + h_i.(T_i^{core})_t) - (P_i^{cache\_hirarchy})_t$
9. **end if**
10. **while** $(\Delta < 0)$
11. $\theta \leftarrow \frac{\Delta}{P_{max}((f_i)_t^{\alpha}/f_{max}^{\alpha})}$
12. $(f_i)_t = \theta.(f_i)_t^*$
13. **Calculate** $J1_{new}$ **and** $(T_i^{core})_{t+1}$
14. **if** $(J1_{new} \langle J1^*$ **and** $(T_i^{core})_{t+1} \rangle T_{max}$ **and** $(f_i)_t > f_{min})$
15. **then**
16. $(f_i)_t^* \leftarrow (f_i)_t$
17. **end if**
18. **for** $k \in [1, N]$ **do:**
19. **if** $(L_{i,k} < L_{i,k}^{max})$ **then:**
20. $L_{i,k}++$
21. **Calculate** $J1_{new}$
22. **if** $(J1_{new} < J1^*$ **and** $(T_i^{top})_t < T_{max})$ **then:**
23. $J1^* = J1_{new}$
24. **else:**
25. $L_{i,k}--$
26. **end if**
27. **end if**
28. **end for**
29. **Calculate** $P_i^{cache\_hirarchy}$ **and** Δ
30. **end while**
31. **end for**
32. **end for**
33.return $\{f^*, L^*\}$

cores selection, cache configuration and frequency assignment that maximizes performance within the dark silicon peak power budget. The overall runtime overhead for this polynomial computation is about 1.16 ms in our experiment.

In this work, since the proposed technique is applied every 100 ms and each time frame is 0.5 ms, then the total number of steps are 200 in each reconfiguration time interval.

References

[1] R.H. Dennard, V.L. Rideout, E. Bassous, A.R. Leblanc, Design of ion-implanted MOSFET's with very small physical dimensions, IEEE J. Solid-State Circuits 9 (5) (1974) 256–268.

[2] H. Esmaeilzadeh, E. Blem, R.S. Amant, K. Sankaralingam, D. Burger, Dark silicon and the end of multicore scaling, in: Proceedings of the International Symposium in Computer Architecture (ISCA), 2011, pp. 365–376.

[3] M.B. Taylor, Is dark silicon useful? Harnessing the four horsemen of the coming dark silicon apocalypse, in: Proceedings of the 49th Annual Design Automation Conference (DAC), 2012, pp. 1131–1136.

[4] P. Bose, Is dark silicon real?: Technical perspective, Commun. ACM Mag. 56 (2) (2013) 92.

[5] J. Kao, S. Narendra, A. Chandrakasan, Subthreshold leakage modeling and reduction techniques, in: Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2002, pp. 141–148.

[6] N.S. Kim, T. Austin, D. Blaauw, T. Mudge, K. Flautner, J.S. Hu, et al., Leakage current: Moore's law meets static power, Computer 36 (12) (2003) 68–75.

[7] W. Wang, P. Mishra, System-wide leakage-aware energy minimization using dynamic voltage scaling and cache reconfiguration in multitasking systems, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 20 (5) (2012) 902–910.

[8] R. Jammy, Materials, process and integration options for emerging technologies, SEMATECH/ISMI Symposium, 2009.

[9] D.H. Woo, N.H. Seong, D.L. Lewis, H.-H.S. Lee, An optimized 3D-stacked memory architecture by exploiting excessive, high-density TSV bandwidth, in: Proceedings of High-Performance Computer Architecture (HPCA), 2010, pp. 1–12.

[10] G. Loh, H Gabriel, 3D-stacked memory architectures for multi-core processors, in: IEEE Computer Society ACM SIGARCH Computer Architecture News, 36, 2008, pp. 453–464.

[11] E. Kultursay, M.T. Kandemir, A. Sivasubramaniam, O. Mutlu, Evaluating STT-RAM as an energy efficient main memory alternative, in: Proceedings of the Performance Analysis of Systems and Software (ISPASS), 2013, pp. 256–267.

[12] B.C. Lee, P. Zhou, J. Yang, Y. Zhang, B. Zhao, E. Ipek, et al., Phase-change technology and the future of main memory, IEEE Micro 30 (1) (2010) 131–141.

[13] G. Venkatesh, J. Sampson, N. Goulding-Hotta, S.K. Venkata, M.B. Taylor, S. Swanson, QsCores: Trading dark silicon for scalable energy efficiency with quasi-specific cores, in: Proceedings of the 44th Annual IEEE/ACM International Symposium on Microarchitecture (Micro), 2011, pp. 163–174.

[14] B. Raghunathan, Y. Turakhia, S. Garg, D. Marculescu, Cherry-picking: Exploiting process variations in dark-silicon homogeneous chip multi-processors, in: Proceedings of the Conference on Design, Automation and Test in Europe (DATE), 2013, pp. 39–44.

[15] Y. Turakhia, B. Raghunathan, S. Garg, D. Marculescu, HaDeS: Architectural synthesis for heterogeneous dark silicon chip multi-processors, in: Proceedings of the 50th Annual Design Automation Conference (DAC), 2013, pp. 1–7.

[16] J. Allred, S. Roy, K. Chakraborty, Designing for dark silicon: a methodological perspective on energy efficient systems, in: Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED), 2012, pp. 255–260.

[17] F. Kriebel, S. Rehman, D. Sun, M. Shafique, J. Henkel, Aser: Adaptive soft error resilience for reliability heterogeneous processors in the dark silicon era, in: Proceedings of the ACM/EDAC/IEEE Design Automation Conference (DAC), 2014, pp. 1–6.

[18] H. Esmaeilzadeh, A. Sampson, M. Ringenburg, L. Ceze, D. Grossman, D. Burger, Addressing dark silicon challenges with disciplined approximate computing, in: Proceedings of the 4th Workshop on Energy-Efficient Design, 2012, pp. 1–2.

[19] H. Esmaeilzadeh, A. Sampson, L. Ceze, D. Burger, Neural acceleration for general-purpose approximate programs, in: Proceedings of the 45th Annual IEEE/ACM International Symposium on Microarchitecture (Micro), 2012, pp. 449–460.

[20] SMP variable, a multi-core CPU architecture for low power and high performance, Whitepaper-http://www.nvidia.com, 2011.

[21] K. Swaminathan, E. Kultursay, V. Saripalli, V. Narayanan, M. Kandemir, S. Datta, Steep slope devices: From dark to dim silicon, IEEE Micro 33 (5) (2013) 50–59.

[22] L. Wang, and K. Skadron, Dark vs. Dim Silicon and Near-Threshold Computing Extended Results, University of Virginia Department of Computer Science Technical Report TR-2013, 1.

[23] U.R. Karpuzcu, A. Sinkar, N.S. Kim, J. Torrellas, Energy smart: Toward energy–efficient many cores for near-threshold computing, in: Proceedings of the High Performance Computer Architecture (HPCA), 2013, pp. 542–553.

[24] G. Venkatesh, J. Sampson, N. Goulding, S. Garcia, V. Bryksin, J. Lugo-Martinez, et al., Conservation cores: Reducing the energy of mature computations, ACM SIGARCH Comput. Archit. News 38 (1) (2010) 205–218.

[25] D. Zhao, H. Homayoun, A.V. Veidenbaum, Temperature aware thread migration in 3D architecture with stacked DRAM, in: Proceedings of the ISQED, 2013, pp. 80–87.

[26] U. Kang, H.J. Chung, S. Heo, D.H. Park, H. Lee, J.H. Kim, et al., 8 gb 3-d ddr3 dram using through-silicon-via technology, IEEE J. Solid-State Circuits 45 (1) (2010) 111–119.

[27] J. Meng, A.K. Coskun, Analysis and runtime management of 3D systems with stacked DRAM for boosting energy efficiency, in: Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE), 2012, pp. 611–616.

[28] H. Tajik, H. Homayoun, N. Dutt, VAWOM: Temperature and process variation aware wearout management in 3D multicore architecture, in: Proceedings of the 50th Annual Design Automation Conference (DAC), 2013, pp. 1–8.

[29] B. Lee, E. Ipek, O. Mutlu, D. Burger, Architecting phase change memory as a scalable DRAM alternative, ACM SIGARCH Comput. Archit. News 37 (3) (2009) 2–13.

[30] Q. Li, Y. He, J. Li, L. Shi, Y. Chen, C.J. Xue, Compiler-assisted refresh minimization for volatile STT-RAM cache, IEEE Trans. Comput. 64 (8) (2015) 2169–2181.

[31] G. Sun, X. Dong, Y. Xie, J. Li, Y. Chen, A novel architecture of the 3D stacked MRAM L2 cache for CMPs, in: Proceedings of the 15th International Symposium on High Performance Computer Architecture (HPCA), 2009, pp. 239–249.

[32] M. Rasquinha, D. Choudhary, S. Chatterjee, S. Mukhopadhyay, S. Yalamanchili, An energy efficient cache design using Spin Torque Transfer (STT) RAM, in: Proceedings of the 16th ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED), 2010, pp. 389–394.

[33] Z. Wang, D.A. Jimenez, C. Xu, G. Sun, Y. Xie, Adaptive Placement and Migration Policy for an STT-RAM-Based Hybrid Cache, in: Proceedings of the High Performance Computer Architecture (HPCA), 2014, pp. 13–24.

[34] S.M. Syu, Y.H. Shao, I.C. Lin, High-endurance hybrid cache design in CMP architecture with cache partitioning and access-aware policy, in: Proceedings of the 23rd ACM International Conference on Great Lakes Symposium on VLSI (GLSVLSI), 2013, pp. 19–24.

[35] Q. Li, J. Li, L. Shi, M. Zhao, C.J. Xue, Y. He, Compiler-assisted STT-RAM-based hybrid cache for energy efficient embedded systems, IEEE Trans. Very Large Scale Integr. (VLSI) Syst., 22 (8) (2014) 829–1840.

[36] W. Kim, M.S. Gupta, G.-Y. Wei, D. Brooks, System level analysis of fast, per–core DVFS using on-chip switching regulators, in: Proceedings of the International Symposium on High Performance Computer Architecture (HPCA), 2008, pp. 123–134.

[37] E. Rotem, A. Mendelson, R. Ginosar, U. Weiser, Multiple clock and voltage domains for chip multi processors, in: Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture (Micro), 2009, pp. 459–468.

[38] T. Chantem, R.P. Dick, X.S. Hu, Temperature-aware scheduling and assignment for hard real-time applications on MPSoCs, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 19 (10) (2011) 1884–1897.

[39] J. Donald, M. Martonosi, Techniques for multicore thermal management: Classification and new exploration, ACM SIGARCH Comput. Archit. News 34 (2) (2006) 78–88.

[40] M. Arora, S. Manne, Y. Eckert, I. Paul, N. Jayasena, D. Tullsen, A comparison of core power gating strategies implemented in modern hardware, ACM SIGMETRICS Perform. Eval. Rev. 42 (1) (2014) 559–560.

[41] A. Kumar, S. Li, P. Li-Shiuan, N.K. Jha, System-level dynamic thermal management for high-performance microprocessors, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 27 (1) (2008) 96–108.

[42] S. Heo, K. Barr, K. Asanović, Reducing power density through activity migration, in: Proceedings of the International Symposium on Low Power Electronics and Design (ISPLED), 2003, pp. 217–222.

[43] J. Henkel, H. Khdr, S. Pagani, M. Shafique, New trends in dark silicon, in: Proceedings of the ACM/EDAC/IEEE Design Automation Conference (DAC), 2015, pp. 1–6.

[44] S. Murali, M. Coenen, A. Radulescu, K. Goossens, G. De Micheli, Mapping and configuration methods for multi-use-case networks on chips, in: Proceedings of Asia and South Pacific Conference on Design Automation (ASP-DAC), 2006, pp. 146–151.

[45] T.C. Xu, G. Schley, P. Liljeberg, M. Radetzki, J. Plosila, H. Tenhunen, Optimal placement of vertical connections in 3d network-on-chip, J. Syst. Archit. 59 (7) (2013) 441–454.

[46] R. Bakker, M.W. Tol, A.D. Pimentel, Emulating asymmetric MPSoCs on the Intel SCC many-core processor, in: Proceedings of 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP), 2014, pp. 520–527.

[47] M. Modarressi, A. Tavakkol, H. Sarbazi-Azad, Virtual point-to-point connections for NoCs, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 29 (6) (2010) 855–868.

[48] A. Asad, M. Seyrafi, A.E. Zonouz, M. Soryani, M. Fathy, A predominant routing for on-chip networks, in: Proceedings of the 4th International Design and Test Workshop (IDT), 2009, pp. 1–6.

[49] Standard performance evaluation corporation [Online], Available: http://www.specbench.org.

[50] A. Hartstein, V. Srinivasan, T.R. Puzak, P.G. Emma, Cache miss behavior: is it $\sqrt{2}$? in: Proceedings of the 3rd Conference on Computing Frontiers, 2006, pp. 313–320.

[51] R. Kumar, G. Hinton, A family of 45 nm IA processors, in: Proceedings of the International Solid-State Circuits Conference-Digest of Technical Papers (ISSCC), 2009, pp. 58–59.

[52] T.S. Muthukaruppan, M. Pricopi, V. Venkataramani, T. Mitra, S. Vishin, Hierarchical power management for asymmetric multi-core in dark silicon era, in: Proceedings of the 50th Annual Design Automation Conference (DAC), 2013, pp. 1–9.
[53] ARM Ltd., http://www.arm.com/products/tools/developmentboards/versatile-express/index.php.
[54] J. Howard, et al., A 48-Core IA-32 message-passing processor with DVFS in 45 nm CMOS, in: Proceedings of the IEEE International Solid-State Circuits Conference (ISSCC), 2010, pp. 108–109.
[55] C. Sun, C.H. Chen, G. Kurian, L. Wei, J. Miller, A. Agarwal, et al., DSENT- a tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling, in: Proceedings of the 6th IEEE/ACM International Symposium on Networks on Chip (NoCS), 2012, pp. 201–210.
[56] M. Chang, P. Rosenfeld, S.L. Lu, B. Jacob, Technology comparison for Large Last-Level Caches (L3Cs): Low-leakage SRAM, Low write-energy STT-RAM, and refresh-optimized eDRAM, in: Proceedings of the 19th International Symposium on High Performance Computer Architecture (HPCA), 2013, pp. 143–154.
[57] Waterloo Maple Software Inc. Maple 9.5 released: 2004.
[58] X. Dong, C. Xu, N. Jouppi, Y. Xie, NVSim: A circuit-level performance, energy, and area model for emerging non-volatile memory, in: Emerging Memory Technologies, Springer, 2012, pp. 15–50.
[59] N. Muralimanohar, R. Balasubramonian, N.P. Jouppi, CACTI 6.0: A Tool to Model Large Caches, HP Laboratories, 2009 Technical Report.
[60] S. Li, J.H. Ahn, R.D. Strong, J.B. Brockman, D.M. Tullsen, N.P. Jouppi, McPAT: an integrated power, area, and timing modeling framework for multicore and manycore architectures, in: Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture (Micro), 2009, pp. 469–480.
[61] M. Palesi, S. Kumar, D. Patti, Noxim: Network-on-chip simulator, 2010. http://noxim.sourceforge.net.
[62] N. Binkert, B. Beckmann, G. Black, S.K. Reinhardt, A. Saidi, A. Basu, et al., The gem5 simulator, ACM SIGARCH Comput. Archit. News 39 (2) (2011) 1–7.
[63] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, M.R. Stan, HotSpot: A compact thermal modeling methodology for early-stage VLSI design, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 14 (5) (2006) 501–513.
[64] A.B. Kahng, B. Li, L.S. Peh, K. Samadi, Orion 2.0: A power-area simulator for interconnection networks, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 20 (1) (2011) 191–196.
[65] T. Zhang, J. Meng, A.K. Coskun, Dynamic cache pooling in 3D multicore processors, ACM J. Emerging Technol. Comput. Syst. (JETC) 12 (2) (2015) 1–20.
[66] J.M. Allred, S. Roy, K. Chakraborty, Dark silicon aware multicore systems: Employing design automation with architectural insight, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 22 (5) (2014) 1192–1196.
[67] J. Zhan, Y. Xie, G. Sun, NoC-Sprinting: Interconnect for fine-grained sprinting in the dark silicon era, in: Proceedings of the 51st ACM/EDAC/IEEE Design Automation Conference (DAC), 2014, pp. 1–6.
[68] C. Bienia, S. Kumar, J.P. Singh, K. Li, The PARSEC benchmark suite: Characterization and architectural implications, in: Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques (PACT), 2008, pp. 72–81.
[69] S. Lee, K. Kang, J. Jung, C.M. Kyung, Runtime 3-D stacked cache data management for energy minimization of 3-D chip-multiprocessors, in: Proceedings of the International Symposium on Quality Electronic Design (ISQED), 2014, pp. 197–204.
[70] K. Kang, J. Jung, S. Yoo, C.M. Kyung, Maximizing throughput of temperature–constrained multi-core systems with 3D-stacked cache memory, in: Proceedings of the International Symposium on Quality Electronic Design (ISQED), 2011, pp. 1–6.
[71] K. Kang, G. De Micheli, S. Lee, C.M. Kyung, Temperature-aware runtime power management for chip-multiprocessors with 3-D stacked cache, in: Proceedings of the International Symposium on Quality Electronic Design (ISQED), 2014, pp. 163–170.
[72] N. Madan, L. Zhao, N. Muralimanohar, A. Udipi, R. Balasubramonian, R. Iyer, S. Makineni, D. Newell, Optimizing communication and capacity in a 3D stacked reconfigurable cache hierarchy, in: Proceedings of the 15th International Symposium on High Performance Computer Architecture (HPCA), 2009, pp. 262–274.
[73] H.Y. Cheng, H-Y, J. Zhan, J. Zhao, Y. Xie, J. Sampson, M.J. Irwin, Core vs. uncore: the heart of darkness, in: Proceedings of the ACM/EDAC/IEEE Design Automation Conference (DAC), 2015, pp. 1–6.
[74] A. Sadeghi, K. Raahemifar, M. Fathy, A. Asad, Lighting the dark-silicon 3D chip multi-processors by exploiting heterogeneity in cache hierarchy, in: Proceedings of the International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC), 2015, pp. 182–186.
[75] M. Monchiero, R. Canal, A. Gonzalez, Power/performance/thermal design-space exploration for multicore architectures, IEEE Trans. Parallel Distrib. Syst. 19 (5) (2008) 666–681.
[76] A. Sodani, Race to exascale: Opportunities and challenges, MICRO 2011 (2011) Keynote talk.
[77] H.Y. Cheng, M. Poremba, N. Shahidi, I. Stalev, M.J. Irwin, M. Kandemir, et al., EECache: A comprehensive study on the architectural design for energy-efficient last-level caches in chip multiprocessors, ACM Trans. Archit. Code Optim. (TACO) 17 (2) (2015) 1–22.
[78] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.